Ref. 2

GATE WAY DEVICE, CLIENT COMPUTER AND DISTRIBUTED FILE SYSTEM CONNECTING THEM

Patent Number: JP10021134
Publication date: 1998-01-23
Inventor(s): DOI KATSUYOSHI; TAKEZAWA SO
Applicant(s): SHARP CORP
Requested Patent: ☐ JP10021134
Application Number: JP19960174561 19960704
Priority Number(s):
IPC Classification: G06F12/00; G06F12/00; G06F3/06; G06F13/00
EC Classification:
Equivalents:

Abstract

PROBLEM TO BE SOLVED: To provide a gate way device improved in a cache hit rate while keeping a cache file fresh.
SOLUTION: This system includes a first proxy process 14 for storing the acquired file object in a cache file 16 and storing an access log 15, a cache updating process 36 for extracting an access log from the access log 15 in accordance with a prescribed rule and a second proxy process 36 for acquiring the file object of the extracted access log from a server computer 11 and storing it in the cache file 16. When the first proxy process 14 is hit to a cache, the cache updating process 36 becomes more effective when second proxy process 34 updates cache contents based on the access log of the file object hit to the cache.

Ref. 2

(19)日本国特許庁(JP)　(12)公開特許公報（A）　(11)特許出願公開番号

特開平１０－２１１３４

(43)公開日　平成10年(1998)1月23日

(51)Int. Cl.[6]		識別記号	庁内整理番号	FI			技術表示箇所
G06F	12/00	545		G06F	12/00	545 A	
		514				514 K	
	3/06	301			3/06	301 M	
	13/00	357			13/00	357 Z	

審査請求　未請求　請求項の数１２　ＯＬ　（全４１頁）

(21)出願番号　特願平8-174561

(22)出願日　平成8年(1996)7月4日

(71)出願人　000005049
シャープ株式会社
大阪府大阪市阿倍野区長池町22番22号

(72)発明者　土居　克良
大阪府大阪市阿倍野区長池町22番22号　シャープ株式会社内

(72)発明者　竹澤　創
大阪府大阪市阿倍野区長池町22番22号　シャープ株式会社内

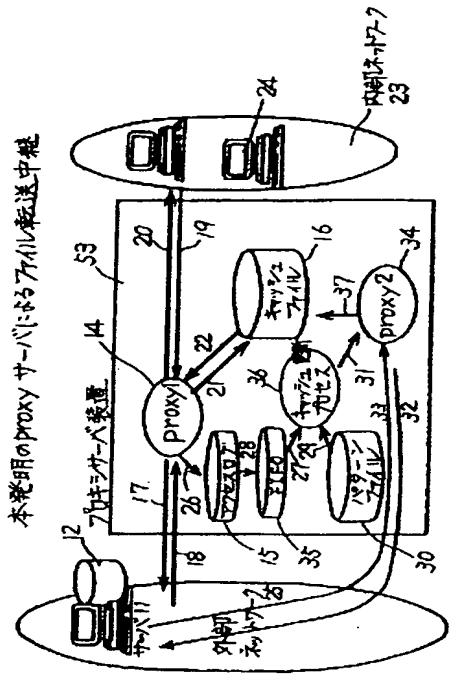
(74)代理人　弁理士　深見　久郎

(54)【発明の名称】ゲートウェイ装置、クライアント計算機およびそれらを接続した分散ファイルシステム

(57)【要約】

【課題】　キャッシュファイルを新鮮に保ちながら、キャッシュヒット率の向上したゲートウェイ装置を提供すること。

【解決手段】　取得したファイルオブジェクトをキャッシュファイル１６に格納し、アクセスログ１５を格納する第１のｐｒｏｘｙプロセス１４に加えて、アクセスログ１５から所定の規則に従ってアクセスログを抽出するキャッシュ更新プロセス３６と、この抽出されたアクセスログのファイルオブジェクトをサーバ計算機１１から取得してキャッシュファイル１６に格納する第２のｐｒｏｘｙプロセス３４とを含む。キャッシュ更新プロセス３６は、第１のｐｒｏｘｙプロセス１４がキャッシュにヒットした場合にそのキャッシュにヒットしたファイルオブジェクトのアクセスログに基づいて第２のｐｒｏｘｙプロセス３４がキャッシュ内容を更新するとさらに効果的である。



1

【特許請求の範囲】

【請求項１】　クライアント計算機が接続される第１のネットワークと、サーバ計算機が接続される第２のネットワークとの間に介在するように設けられるゲートウェイ装置であって、

前記第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトにより指定される前記第２のネットワークのサーバ計算機に対して、当該ファイルオブジェクトに対するアクセス要求を行なうネットワークファイル中継手段を含み、

前記ネットワークファイル中継手段は、

取得したファイルオブジェクトを、所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、

過去の一定期間内に行なわれたファイルオブジェクトの転送記録を蓄積するファイル転送記録手段と、

ファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在する場合は当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在しない場合には当該ファイルオブジェクトにより指定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含み、

前記ゲートウェイ装置はさらに、前記ファイル転送記録手段から所定の規則に従って転送記録を抽出し、当該転送記録のファイルオブジェクトを前記第２のネットワークのサーバ計算機から取得して前記キャッシュファイル手段に格納するためのキャッシュ更新手段を含むゲートウェイ装置。

【請求項２】　前記キャッシュ更新手段はさらに、特定の文字列パターンを予め記憶するための文字列パターン記憶手段を含み、

前記キャッシュ更新手段は、前記特定の文字列パターンと一致するファイルオブジェクト名称を含む転送記録を前記ファイル転送記録手段から抽出し、当該ファイルオブジェクトを前記第２のネットワークのサーバ計算機から取得して前記キャッシュファイル手段に格納する、請求項１記載のゲートウェイ装置。

【請求項３】　前記ネットワークファイル中継手段は、各々が第１のネットワークのクライアント計算機からのファイルオブジェクトのアクセス要求を処理するための第１の転送プロセスを複数個、同時並列的に起動させ、

前記キャッシュ更新手段は、各々が１つの前記ファイルオブジェクトの取得を行なうための第２の転送プロセスを複数個、同時並列的に起動させ、前記第１の転送プロセスの数を検知して、前記第１の転送プロセスの数と前記第２の転送プロセスの数との和が予め定められる上限以下となるように前記第２の転送プロセスの新たな起動を制御する、請求項１記載のゲートウェイ装置。

2

【請求項４】　前記キャッシュ更新手段は、前記ファイル転送記録手段の中から、キャッシュにヒットしたファイルオブジェクトの転送記録を抽出し、当該ファイルオブジェクトを前記第２のネットワークのサーバ計算機から取得して前記キャッシュファイル手段に格納する、請求項１記載のゲートウェイ装置。

【請求項５】　前記キャッシュ更新手段はさらに、前記ファイル転送記録手段の末尾から転送記録を順次読出して格納し、先に格納された転送記録から順に出力するＦＩＦＯバッファ手段を含み、

前記キャッシュ更新手段は、前記ＦＩＦＯバッファ手段から転送記録を読出して、当該転送記録のファイルオブジェクトを前記第２のネットワークのサーバ計算機から取得して前記キャッシュファイル手段に格納する、請求項１記載のゲートウェイ装置。

【請求項６】　前記キャッシュ更新手段は、前記ファイル転送記録手段から抽出した転送記録のファイルオブジェクトの前回の更新が所定時間以内であれば、前記第２のネットワークのサーバ計算機へのアクセスを行なわない、請求項１記載のゲートウェイ装置。

【請求項７】　クライアント計算機と第１のネットワークによって接続され、かつサーバ計算機と第２のネットワークによって接続される複数のキャッシュサーバ計算機と第３のネットワークによって接続されるゲートウェイ装置であって、

前記第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行なうための演算手段と、

当該演算結果に基づいて前記第３のネットワークの複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択するための中継制御手段とを含むゲートウェイ装置。

【請求項８】　クライアント計算機とゲートウェイ装置とが第１のネットワークで接続され、サーバ計算機とキャッシュサーバ計算機とが第２のネットワークで接続され、複数の前記ゲートウェイ装置と複数の前記キャッシュサーバ計算機とが第３のネットワークで接続される分散ファイルシステムであって、

前記ゲートウェイ装置は、前記第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行なうための演算手段と、

当該演算結果に基づいて前記第３のネットワークの複数のキャッシュサーバ計算機の中からアクセス要求を出力

するキャッシュサーバ計算機を選択するための中継制御手段とを含む分散ファイルシステム。

【請求項９】　クライアント計算機が接続される第１のネットワークと、サーバ計算機が接続される第２のネットワークとの間に介在するように設けられるゲートウェイ装置であって、

前記第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、該演算結果に基づいてゲートウェイ装置を選択し、該選択が他のゲートウェイ装置であれば、当該アクセス要求を第３のネットワークを介して前記他のゲートウェイ装置へ出力するための中継制御手段と、

前記第２のネットワークのサーバ計算機から取得したファイルオブジェクトを、所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、

過去の、一定時間内に行なわれたファイルオブジェクトの転送記録を蓄積するためのファイル転送記録手段と、

前記中継制御手段が自身のゲートウェイ装置を選択した場合と前記第３のネットワークを介して他のゲートウェイ装置からアクセス要求を受信した場合に、ファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在する場合は当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在しない場合には、当該ファイルオブジェクトにより指定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含むゲートウェイ装置。

【請求項１０】　サーバ計算機と複数のキャッシュサーバ計算機とが第２のネットワークで接続され、前記複数のキャッシュサーバ計算機と第１のネットワークによって接続されるクライアント計算機であって、

キャッシュサーバ計算機に対するファイルオブジェクトのアクセス要求を出力し、当該ファイルオブジェクトを取得するためのファイル取得手段と、

キャッシュサーバ計算機へのファイルオブジェクトに対するアクセス要求を出力する際、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、当該演算結果に基づいて前記複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択するための中継制御手段とを含むクライアント計算機。

【請求項１１】　前記中継制御手段はさらに、第３のネットワークを介して他のクライアント計算機からのファイルオブジェクトに対するアクセス要求を受信するための受信手段を含み、

前記中継制御手段は、前記他のクライアント計算機からのアクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、当該演算結果に基づいて前記複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択する、請求項１０記載のクライアント計算機。

【請求項１２】　クライアント計算機と第１のネットワークによって接続され、かつサーバ計算機と第２のネットワークによって接続される複数のキャッシュサーバ計算機と第３のネットワークによって接続されるゲートウェイ装置であって、

前記第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、該演算結果に基づいてゲートウェイ装置を選択し、該選択が他のゲートウェイ装置であれば、当該アクセス要求を第３のネットワークを介して前記他のゲートウェイ装置へ出力するための中継制御手段と、

前記第２のネットワークのサーバ計算機から取得したファイルオブジェクトを、所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、

過去の、一定時間内に行なわれたファイルオブジェクトの転送記録を蓄積するためのファイル転送記録手段と、

前記中継制御手段が自身のゲートウェイ装置を選択した場合と前記第３のネットワークを介して他のゲートウェイ装置からアクセス要求を受信した場合に、ファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在する場合は当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトが前記キャッシュファイル手段に存在しない場合には当該ファイルオブジェクトにより指定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含むゲートウェイ装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ネットワーク上に分散した複数のサーバ計算機と、複数のクライアント計算機とがゲートウェイ装置（計算機）を介して通信回線で相互に接続されている分散ファイルシステムに関し、特に、複数のクライアント計算機がゲートウェイ装置を介してサーバ計算機内部の記憶装置内のファイルオブジェクトにアクセスする場合の分散ファイルシステムのフ

5

ファイル中継を行なうゲートウェイ装置、ゲートウェイ装置にアクセス要求を出力するクライアント計算機、およびそれらを接続した分散ファイルシステムに関する。

【０００２】

【従来の技術】以下の説明で「サーバ計算機」とは、ネットワーク上において何らかのサービスを提供している計算機をいい、「クライアント計算機」とは、そうしたサーバ計算機のサービスを要求し、受ける計算機をいうものとする。また「ファイルオブジェクト」とは、分散ファイルシステムの利用するネットワークプロトコルと、ネットワークアドレス（サーバ計算機の名称）と、ファイル名称と、ファイルの実体との組をいうものとする。ここで「ファイルオブジェクトの名称」とは、ネットワークプロトコルとネットワークアドレスとファイル名称との組のことをいう。

【０００３】従来、上述したような分散ファイルシステムにおいては、複数のクライアント計算機からのサーバ計算機上のファイルオブジェクトに対する読出要求は、一旦途中のゲートウェイ計算機で中継されていた。ゲートウェイ計算機がサーバ計算機から読出したファイルオブジェクトは、クライアント計算機に中継される。それと同時にそれらファイルオブジェクトは、ゲートウェイ計算機内部のキャッシュファイル（固定ディスク、半導体メモリなど）に蓄積される。そうしたゲートウェイ計算機の一例が特開平４－３１３１２６号公報（発明の名称「分散ファイルシステムのファイル入出力方式」）に開示されている。

【０００４】図１５は、この従来の分散ファイルシステムを示している。このシステムは、ゲートウェイ計算機１１０と、このゲートウェイ計算機１１０を介して相互に接続されたサーバ計算機１００および複数のクライアント計算機１２０を含む。サーバ計算機１００は、複数のクライアント計算機１２０によって共有されている。

【０００５】サーバ計算機１００は、ファイルを格納するディスク装置１３６と、ネットワークを介してディスク装置１３６のファイルの内容を入出力するためのファイル入出力応答手段１３４と、ディスク装置１３６内のブロック情報をネットワークを介してクライアント計算機１２０に応答するためのブロック情報応答手段１３２とを含む。

【０００６】クライアント計算機１２０は、サーバ計算機１００のブロック情報応答手段１３２に対してブロック情報要求を送り、ブロック情報を受けるためのブロック情報要求手段１５２と、ゲートウェイ計算機１１０を介してサーバ計算機１００のファイル入出力応答手段１３４に対してファイル入出力要求を送り、当該ファイルを受信するためのファイル入出力要求手段１５４とを含む。

【０００７】ゲートウェイ計算機１１０は、クライアント計算機１２０（複数）とサーバ計算機１００との間に介在し、サーバ計算機１００内のディスク装置１３６と、各クライアント計算機１２０との間のキャッシュメモリとして機能するものであり、キャッシュした内容を格納するためのディスクキャッシュ１４４と、ファイル入出力要求手段１５４およびファイル入出力応答手段１３４の間に介在するキャッシュ管理手段１４２とを含む。

6

【０００８】このシステムは次のように動作する。まず、クライアント計算機１２０がサーバ計算機１００内のディスク装置１３６内のファイルにアクセスして１つのブロックを読込む場合の動作について説明する。ブロック情報要求手段１５２は、ゲートウェイ計算機１１０を介してサーバ計算機１００に対して読込対象のブロックにかかるブロック情報の取得を要求するブロック情報要求メッセージを発行する。

【０００９】サーバ計算機１００内のブロック情報応答手段１３２は、このメッセージに応答して、ディスク装置１３６から該当するブロック情報を取出し、そのブロック情報を有するブロック情報応答メッセージをゲートウェイ計算機１１０を介してクライアント計算機１２０内のブロック情報要求手段１５２に返却する。

【００１０】クライアント計算機１２０のファイル入出力要求手段１５４は、返却されたブロック情報に基づいて読込対象のブロックの読込を要求するファイルアクセス要求を、ゲートウェイ計算機１１０内のキャッシュ管理手段１４２に対して発行する。

【００１１】このファイルアクセス要求を受取ったキャッシュ管理手段１４２は、そのファイルアクセス要求にかかるブロック情報とディスクキャッシュ１４４に記憶されている読込対象のブロックにかかるブロック情報との比較を行なう。そしてディスクキャッシュ１４４に、該当するブロック情報が存在しない場合、または両方のブロック情報の内容（更新時間）が異なる場合（読込対象のブロックのキャッシュが有効でない場合）には、キャッシュ管理手段１４２は上述のファイルアクセス要求をサーバ計算機１００のファイル入出力応答手段１３４に対して発行する。

【００１２】サーバ計算機１００のファイル入出力応答手段１３４は、このファイルアクセス要求に応答して、ディスク装置１３６内のファイルをアクセスして、該当するブロックを読込み、ゲートウェイ計算機１１０にそのブロックを転送する。

【００１３】キャッシュ管理手段１４２は、このブロックをディスクキャッシュ１４４に格納し、そのブロックにかかるブロック情報のディスクキャッシュ１４４への設定または更新を行なう。キャッシュ管理手段１４２は同時に、そのブロックをクライアント計算機１２０のファイル入出力要求手段１５４に対して転送する。

【００１４】以上でディスクキャッシュ１４４内に有効なファイルが存在しない場合のこのシステムの動作につ

いて説明した。

【0015】次回に同一ファイルに対するアクセス要求があった場合には、ゲートウェイ計算機110のキャッシュ管理手段142は、ディスクキャッシュ144から当該ブロックを取出し、アクセス要求を発行したクライアント計算機120のファイル入出力要求手段154に対して当該ブロックを転送する。したがってこの場合、ゲートウェイ計算機110からサーバ計算機100に対する転送要求は発行されない。したがって中継アクセスの高速化が図られる。

【0016】類似の技術が、特開昭63－200244号公報（発明の名称「ファイル・アクセス・システム」）や、特開昭63－20184号公報（発明の名称「遠隔ファイル・アクセス・システム」）にも開示されている。いずれの技術においても、あるファイルオブジェクトに対するアクセス要求が発行された場合に、当該ファイルオブジェクトの内容が更新されているかどうかを確認するために、サーバ計算機の保持するファイルオブジェクトの変更時刻をキャッシュファイルの変更時刻と比較する。そしてキャッシュファイルの内容が古い場合には、クライアント計算機は更新されたサーバ計算機の内容を読出すようになっている。しかし、これら2件の公報に記載された技術では、キャッシュファイルはクライアント計算機内に存在しており、特開平4－313126号公報に開示されたシステムのようにシステム内にゲートウェイ計算機が含まれているわけではない。

【0017】しかし、これら3件の従来技術文献は、サーバ計算機のファイルオブジェクトと、キャッシュされたファイルオブジェクトとの内容が食い違わないような工夫をしているという共通点を持っている。

【0018】このように本来同一であるべきファイルの食い違いを防ぐことをインテグリティ（Integrity）を保つという。また、食い違いの起きたファイルのデータをステールデータ（stale data）と呼ぶ。

【0019】上述した従来の技術の基本は、ネットワーク経由でサーバ計算機の、該当するファイルオブジェクトの変更時刻を調べ、対応するキャッシュファイルの内容の変更時刻と比較するアルゴリズムを使用している。そのため、サーバ計算機に対するネットワーク経由の問合せを必ず行なわなければならない。ネットワークの実効的な転送速度が小さい場合や、該当するファイルオブジェクトを有するサーバ計算機にかかっている負荷が高くて即座に応答できない場合には、そうした問合せ自体にかなり時間がかかってしまうことがある。そのためネットワークファイルシステムによっては、このようなファイルキャッシュのシステムを採用していない場合もある。すなわちこうしたシステムでは、サーバ計算機の保持するファイルオブジェクトが更新されているにもかかわらず、ゲートウェイ計算機の持つキャッシュ内の対応するファイルオブジェクトは必ずしも更新されていない。このようなネットワークファイルシステム方式の代表的なものとして、いわゆるインターネットにおけるHTTP（Hyper Text Transfer Protocol）を使った広域分散型マルチメディア情報システムがある。

【0020】インターネットは、その基本プロトコルとしてTCP／IPプロトコルを利用した、グローバルなネットワークである。インターネット上に構築された地域分散型マルチメディア情報提供システムは、World Wide Wed（WWW）と呼ばれる。WWWは、ネットワーク上に分散したファイルオブジェクトを扱うことができる。これらのファイルオブジェクトは、単なるテキストデータにとどまらず、画像データ、音声データ、ビデオ画像データなどさまざまな種類のものを含む。WWWは、情報提供側にとっても、情報利用者（ユーザ）にとっても魅力的であるため、ネットワーク上のWWWに関するトラフィックが爆発的に増加している。WWWシステムでは、クライアント計算機のユーザは、グラフィカルユーザインタフェースを持ったブラウザソフトウェアをクライアント計算機上で実行させるだけで、ネットワーク上に分散したサーバ計算機の保持するさまざまなファイルオブジェクトで構成された情報を次々とアクセスすることができる。WWWは、このような操作の簡単さのために近時めざましい普及をみせている。

【0021】そしてこのWWWシステムは、TCP／IPプロトコルの上に構築されたHTTPプロトコルでファイルオブジェクトを転送している。

【0022】HTTPプロトコルを実施する上においては、特開平4－313126号公報に記載されたようなゲートウェイ計算機によりファイルオブジェクトを中継転送することが広く行なわれている。データは、ゲートウェイ装置にキャッシュされる。またWWWシステムでは、ファイルオブジェクトの中継転送においてステールデータが発生することは容認されている。すなわちゲートウェイ計算機にキャッシュされている内容が、サーバ計算機の当該ファイルオブジェクトと一致していない場合を許容している。

【0023】キャッシュファイル内のステールデータの率をステールデータ率と呼ぶこととする。何らかの方式でキャッシュファイルの内容を更新しない限り、ステールデータ率は増加する。一般にキャッシュファイルのステールデータ率が増加すると、クライアント計算機を操作しているユーザは、内容が古いと自主的に判断して、ファイルオブジェクトをサーバ計算機から再度ロードすることを試みる。この場合、キャッシュファイルの内容を利用しないプロトコルが用いられる。この結果、ゲートウェイ計算機内のキャッシュファイルシステムが意味を持たなくなる傾向がある。

【0024】一方で、このようなステールデータの発生を容認するプロトコルの利点もある。クライアント計算機が、サーバ計算機のファイルオブジェクトのアクセス

をゲートウェイ計算機に中継依頼してアクセスを行なった場合、ゲートウェイ計算機のキャッシュファイルにヒットした場合には、キャッシュファイルからファイルオブジェクトが読出されてクライアント計算機に転送される。サーバ装置に対する当該ファイルオブジェクトの変更時刻の問合せは不要である。そのため短時間にファイルオブジェクトがキャッシュから取出され、クライアントに返送される。

【００２５】ＨＴＴＰプロトコルのようにステールデータを容認するネットワークファイルプロトコルは、全地球的な広域ネットワークで利用される上で有利な点を有している。すなわち、ゲートウェイ装置のキャッシュに当該ファイルオブジェクトがキャッシュされていれば、サーバ装置に対するアクセスが発生しない。その結果応答時間短縮を図ることができるという有利な点を持っている。

【００２６】上述のようにネットワーク上のファイルオブジェクトのアクセスを高速化し、かつインターネットのトラフィックを低減させるゲートウェイ計算機は特に「ｐｒｏｘｙサーバ装置」とも呼ばれている。

【００２７】「ｐｒｏｘｙ」とは「代理人」という意味である。ｐｒｏｘｙサーバ装置は、その名のとおり、クライアント計算機がネットワーク上のサーバ計算機のファイルオブジェクトにアクセスする場合に、そのアクセス要求をサーバ計算機を代理して受付け、ファイルオブジェクトの転送を中継する機能を果たす。ｐｒｏｘｙサーバ装置の概念を図１６を参照して説明する。

【００２８】ｐｒｏｘｙサーバ装置１３は、たとえばある企業内に用いられた内部ネットワーク２３と、その企業外の外部ネットワーク（インターネットなど）２５との間に介在して設けられるものである。内部ネットワーク２３は複数個のクライアント計算機２４を含み、外部ネットワーク２５は、同じく複数個のサーバ計算機１１を含んでいる。各サーバ計算機１１は、ファイルオブジェクト１２を保有している。

【００２９】ｐｒｏｘｙサーバ装置１３は、キャッシュファイル１６と、クライアント計算機からリード要求１９を受け、キャッシュファイル１６内に当該ファイルオブジェクトが存在する場合にはそのファイルオブジェクトをデータ２０として返送し、キャッシュファイル１６内に当該ファイルオブジェクトの有効なものが存在しない場合には外部ネットワーク２５を介してサーバ計算機１１に対するリード要求１７を発行し、対応するファイルオブジェクトをデータ１８として受けてクライアント計算機２４に対して返送するｐｒｏｘｙプロセス１４と、ｐｒｏｘｙプロセス１４によるファイルオブジェクトの転送記録を記録するアクセスログ１５とを含む。ｐｒｏｘｙプロセス１４は、サーバ計算機１１からファイルオブジェクトを取得したときには、当該ファイルオブジェクトをキャッシュファイル１６にも新たに書込む機能を有している。

【００３０】このｐｒｏｘｙサーバ装置１３は次のように動作する。まず、内部ネットワーク２３内のクライアント計算機２４が、外部ネットワーク２５上のサーバ計算機１１のファイルオブジェクト１２に対するリード要求１９をｐｒｏｘｙサーバ装置１３内のｐｒｏｘｙプロセス１４に対して発行する。ｐｒｏｘｙプロセス１４はこのリード要求１９に応答して、キャッシュファイル１６をアクセスし、キャッシュファイル中に当該ファイルオブジェクトのキャッシュデータがあるか否かを判定する。当該ファイルオブジェクトがキャッシュされていれば、ｐｒｏｘｙサーバ装置１３固有のキャッシュファイル有効期限と、キャッシュされたファイルオブジェクトの最終変更時刻とを比較する。キャッシュが有効期限内であれば、キャッシュファイルからファイルオブジェクト読出（２２）、そのデータ２０を内部ネットワーク２３のリード要求１９を発行したクライアント計算機２４に対して返送する。

【００３１】キャッシュファイル１６内に該当ファイルオブジェクトが存在しない場合、および存在していたとしても有効期限が過ぎている場合には、キャッシュファイル１６内に有効なファイルオブジェクトが存在しないものと判定される。その場合、ｐｒｏｘｙプロセス１４は、外部ネットワーク２５のサーバ計算機１１に対して、ファイルオブジェクト１２の読出要求１７を発行する。これに応答してサーバ計算機１１は、ファイルオブジェクト１２をデータ１８としてｐｒｏｘｙプロセス１４に返送する。ｐｒｏｘｙプロセス１４は、ログファイル１５にアクセスログとして、ファイルオブジェクト名称（すなわちネットワークプロトコル名称と、サーバ計算機のネットワークアドレス名称と、ファイル名称との組）と、ファイルオブジェクトのデータの実体（すなわちファイルオブジェクトそのもの）と、書込時刻とを書込む（２６）。

【００３２】ｐｒｏｘｙプロセス１４はさらに、内部ネットワーク２３内のクライアント計算機２４にデータ２０を転送し、かつキャッシュファイル１６に、当該ファイルオブジェクトを書込む（２１）。

【００３３】したがってキャッシュファイル１６内には、ファイルオブジェクト名称と、ファイルオブジェクトの実体と、ファイルオブジェクトをサーバ装置から取得した時刻（最終変更時刻）とが記録されている。

【００３４】ｐｒｏｘｙサーバ装置は、ヨーロッパ原子核物理研究所ＣＥＲＮで開発されたｈｔｔｐｄや、日本の電子技術総合研究所の佐藤豊氏の開発されたＤｅｌｅＧａｔｅなどのソフトウェアを、ネットワーク接続されたＵＮＩＸの動くコンピュータ上で実行することにより実現されることが一般的である。

【００３５】ｐｒｏｘｙサーバ装置の物理構成を図１７に示す。ｐｒｏｘｙサーバ装置は、ＵＮＩＸの動作する

ワークステーション２００により構成される。ワークステーション２００は、ＣＰＵ（中央処理装置）２０２と、ＣＰＵ２０２に対して内部バス２０４により接続されるメモリ２０６と、ファイル用のＩ／Ｏ（入出力）装置２０８と、ルータ２１０を介して外部ネットワークに接続される第１ネットワークＩ／Ｏインタフェース２１２と、内部ネットワークに接続される第２ネットワークＩ／Ｏインタフェース２１４とを含む。Ｉ／Ｏ装置２０８には、キャッシュファイルの蓄積、アクセスログ、および各種ワークファイルの記憶場所として使用されるファイル部２１６が接続されている。

【００３６】このワークステーション２００は、ＣＰＵ２０２が、上述したｈｔｔｐｄやＤｅｌｅＧａｔｅというソフトウェアを実行することによりｐｒｏｘｙサーバ装置１３を実現する。

【００３７】図１８は、ｐｒｏｘｙサーバ装置の他の構成例を示す図である。このｐｒｏｘｙサーバ装置はワークステーション３００からなっている。ワークステーション３００は、概略、図１７のワークステーション２００と同様の構成であるが、図１７の第２ネットワークＩ／Ｏインタフェース２１４に代えて、モデム装置３０４に接続されるシリアルポート３０２を有している点が異なっている。そしてこのワークステーション３００は、モデム装置３０４および公衆電話回線網３０６を介して内部ネットワークに接続されている。

【００３８】図１８において、図１７と同一の部品には同一の参照番号を付している。それらの名称および機能も同一である。したがって、ここではそれらについての詳しい説明は繰返さない。

【００３９】図１９にｐｒｏｘｙサーバ装置のさらに他の構成例を示す。図１９のｐｒｏｘｙサーバ装置も、同じくワークステーション４００により実現される。このワークステーション４００は、概略、図１７に示されるワークステーション２００と同様の構成であるが、第１ネットワークＩ／Ｏインタフェース２１２および第２ネットワークＩ／Ｏインタフェース２１４に代えて、内部ネットワーク２３に接続された１つのネットワークＩ／Ｏインタフェース４０２を有している点で異なっている。内部ネットワーク２３は、ルータ２１０を介して外部ネットワーク２５に接続されており、また複数個のクライアント計算機２４にも接続されている。ワークステーション４００は、内部ネットワーク２３およびルータ２１０を介して外部ネットワーク２５のサーバ計算機１１と通信するとともに、内部ネットワーク２３およびクライアント計算機２４を介してユーザ３１と通信するこ＊とができる。

【００４０】図１９と図１７とにおいて、同一の部品には同一の参照符号および名称を与えている。それらの機能も同一である。したがって、ここではそれらについての詳しい説明は繰返さない。

【００４１】ｐｒｏｘｙサーバ装置は、既に述べたように中継データのキャッシュ機構を有する。すなわち、ｐｒｏｘｙサーバ装置はその内部に存在するキャッシュファイル装置に、中継するファイルオブジェクトのデータをキャッシュする。

【００４２】ｐｒｏｘｙサーバ装置のキャッシュファイルには、ｐｒｏｘｙサーバ装置固有の有効期限が定められている。ファイルオブジェクトがキャッシュに書込まれてから有効期限内に、同じサーバ計算機上のファイルオブジェクトへのアクセス要求をクライアント計算機から受取った場合には、キャッシュファイルに以前に書込まれたファイルオブジェクトを取出してクライアント計算機に転送する。これによりサーバ計算機に対するアクセスは発生しない。

【００４３】このようなゲートウェイ装置によるキャッシュの効果を以下具体的に説明する。ｐｒｏｘｙサーバ装置はある会社組織に設置し、外部のネットワークとは６４ｋｂｐｓの速度で接続するものとする。また、このｐｒｏｘｙサーバ装置は一方で、１０Ｍｂｐｓの速度の会社組織内のローカルエリアネットワークに接続されるものとする。外部と内部のネットワークの速度差は１桁以上ある。このような例はよく見られる例であり、その典型的な例が図１９に示す構成である。

【００４４】このとき、ｐｒｏｘｙサーバ装置で中継ファイルオブジェクトをキャッシュするものとして、キャッシュのヒット率をＨｉｔｒａｔｅとする。Ｈｉｔｒａｔｅは、ファイルオブジェクトに対するアクセス要求に対してキャッシュが１００％ヒットするとき１であり、０％ヒットするとき０になる指数である。

【００４５】なお、ｐｒｏｘｙサーバ装置におけるキャッシュファイルからのデータ読出およびキャッシュファイルへのデータ書込速度は、ネットワーク速度に比べて十分高速であるものとする。すなわち、キャッシュファイルに対するファイルオブジェクトの入出力に要する時間は無視できるものとする。

【００４６】このとき、外部ネットワークにあるファイルオブジェクトを内部からアクセスするときの平均アクセス速度Ｖａｖｅは次の数１によって表わされる。

【００４７】

【数１】

$$Vave(kbps) = 64kbps \times (1 - Hitrate) + 10000kbps \times Hitrate$$

【００４８】キャッシュファイルが存在しない場合、Ｈｉｔｒａｔｅは０である。したがって平均速度（ｋｂｐｓ）は、数１から外部ネットワークのアクセス速度６４ｋｂｐｓと等しくなる。また、Ｈｉｔｒａｔｅが１であれば、すなわち１００％キャッシュファイルにヒットするのであれば、平均速度（ｋｂｐｓ）は、内部ネットワークの速度と等しく１０，０００ｋｂｐｓ（＝１０Ｍｂｐｓ）となる。つまり、ヒット率が高いほどクライアン

ト計算機から見た平均アクセス速度は高くなる。ヒット率と平均アクセス速度との関係を示す数１をグラフ化したものを図２０に示す。

【００４９】ところで、キャッシュヒット率は、キャッシュファイルの量に依存している。すなわち、ファイルオブジェクトをなるべくたくさんキャッシュファイルの中に蓄積しておけば、再びアクセスされたファイルオブジェクトやキャッシュファイル中に含まれる確率は高く、ヒット率は高くなる。そうした関係を示すものとして、キャッシュヒット率と、キャッシュファイルの蓄積量との関係を図２１にグラフとして示す。図２１に示すように、キャッシュファイルサイズを小さくすればキャッシュヒット率は０に近く、キャッシュファイルサイズを大きくすればキャッシュヒット率は徐々に上昇する。

【００５０】ｐｒｏｘｙプロセスにおいては、キャッシュファイルの内容を維持する方式として、最初のファイルアクセスによる書込が起きてから一定期間経過したファイルオブジェクトを無効としていく制御方式が取られている。この一定期間を有効期限と呼んでいる。この有効期限は、ｐｒｏｘｙサーバ装置を管理する者が実情に合わせて適切な値を設定する。有効期限を長く取れば、キャッシュファイルに蓄積されるデータ量は増加する。その結果、キャッシュのヒット率は高くなる。すなわち、キャッシュのヒット率を上げるには、キャッシュの有効期限を長く取ればよい。

【００５１】なお、キャッシュファイルを格納するファイル装置の記憶容量は有限であるから、有効期限を無制限に長くするわけにはいかない。有効期限が過ぎたキャッシュファイルは、何らかのタイミングで消去される。このようにキャッシュファイルを消去するまでの時間を、この明細書では消去時間と呼ぶことにする。

【００５２】本願発明者の実験によれば、キャッシュ有効期限とキャッシュファイルに蓄積されるファイル量とほぼ比例する。この関係を図２２に示す。これは、内部ネットワークから外部ネットワークへのアクセス量が、日によらずほぼ一定であるためである。したがって次のような関係が成り立つ。

【００５３】

【数２】

$$\text{キャッシュ有効期限} \propto \text{キャッシュファイル蓄積量} \propto \text{キャッシュヒット率}$$

【００５４】この関係から、図２２に示したように、キャッシュ有効期限を増やすとキャッシュのヒット率が高まることがわかる。

【００５５】キャッシュ有効期限とファイルオブジェクトの量との関係は、クライアント計算機を利用する内部ユーザが、外部ネットワーク上のサーバ計算機のファイルオブジェクトをアクセスする頻度によって異なる。キャッシュ有効期限とキャッシュファイル蓄積量とは、ほぼ比例するものの、その比例係数はネットワーク速度、内部クライアント計算機ユーザ数などの使用環境により変動する。

【００５６】なお、キャッシュファイル１日程度の有効期限では、キャッシュヒット率は１２％程度である。有効期限を１４日にするとキャッシュヒット率は４１％程度になることが経験されている。有効期限１４日のときにはファイルオブジェクトが７００ＭＢ程度キャッシュファイルに蓄積されていることが観察されている。

【００５７】また、近年ローカルネットワークの規模が大きくなり、接続されるクライアント計算機の数が増大するにつれ、またはクライアント計算機の読出要求が増大するにつれて、単一のｐｒｏｘｙサーバ計算機では、各クライアント計算機からの要求に十分に応えることが難しくなってきている。そこで、複数のｐｒｏｘｙサーバ計算機を組合せて、ファイルオブジェクトに対するアクセス処理の効率の向上を図る方式が提案されており実施されている。

【００５８】図２３は、複数のｐｒｏｘｙサーバ計算機を用いたシステムの構成例である。ｐｒｏｘｙサーバ計算機５０１は、広域ネットワークとローカルネットワーク間の中継を行ない、ファイルオブジェクトをキャッシュする。このｐｒｏｘｙサーバ計算機５０１は、以後２次キャッシュｐｒｏｘｙサーバ計算機と呼ぶ。これに対して、ｐｒｏｘｙサーバ計算機５０３～５０８は、１次キャッシュｐｒｏｘｙサーバ計算機と呼ばれ、さらに分割されたローカルネットワークに接続されるクライアント計算機２４と２次キャッシュｐｒｏｘｙサーバ計算機５０１間の中継およびファイルオブジェクトのキャッシュを行なう。

【００５９】たとえば、クライアント計算機２４から１次キャッシュｐｒｏｘｙサーバ計算機５０４に対してファイルオブジェクトに対する読出要求を出力したとする。このとき、１次キャッシュｐｒｏｘｙサーバ計算機５０４のキャッシュファイル５１０内に有効なファイルオブジェクトが存在する場合、１次キャッシュｐｒｏｘｙサーバ計算機５０４は、キャッシュファイル５１０よりファイルオブジェクトを取出し、読出要求を出したクライアント計算機２４に送信する。

【００６０】また、１次キャッシュｐｒｏｘｙサーバ計算機５０４に有効なファイルオブジェクトが存在しなかった場合、クライアント計算機２４からの読出要求は２次キャッシュｐｒｏｘｙサーバ計算機５０１に中継される。２次キャッシュｐｒｏｘｙサーバ計算機５０１は、同様にキャッシュファイル５０２内の有効なファイルオブジェクトの有無をチェックし、当該ファイルオブジェクトが存在すれば１次キャッシュｐｒｏｘｙサーバ計算機５０４に送出し、存在しなければ読出要求を広域ネットワーク６１を介してサーバ計算機１１に中継する。

【００６１】以降の処理は、単体のｐｒｏｘｙサーバ計

15

算機と同様であるので、ここではその詳しい説明は繰返さない。

【００６２】この方式によって、各々の１次キャッシュｐｒｏｘｙサーバ計算機５０３～５０８は、クライアント計算機を分担して受け持つことができ、負荷の分散が図れる。また、２次キャッシュｐｒｏｘｙサーバ計算機５０１も、各１次キャッシュｐｒｏｘｙサーバ計算機５０３～５０８でキャッシュされていなかったファイルオブジェクトに関してのみ処理を行なえばよいので、負荷＊

16

＊の軽減が図れる。具体的には、全体でＮ台のクライアント計算機２４が存在する場合、従来の方式ではｐｒｏｘｙサーバ計算機は、Ｎ台のクライアント計算機の要求を処理する必要があったが、この方式では数３で算出される台数分のクライアント計算機からの要求を処理すればよいことになる。

【００６３】

【数３】

$$\left(1-\left(\text{一次キャッシュproxyサーバ計算機のキャッシュヒット率}\right)\right)\times N\text{台}$$

【００６４】しかしながら、この方式では１台の２次キャッシュｐｒｏｘｙサーバ計算機５０１が広域ネットワーク６１に接続されるサーバ計算機１１との中継をすべて行なっており、本質的な負荷の分散が達成されているとは言い難い。

【００６５】また、キャッシュファイルの使用にも無駄が多い。すなわち、１次キャッシュｐｒｏｘｙサーバ計算機群５０３～５０８の有するファイルオブジェクトの内容はすべて２次キャッシュｐｒｏｘｙサーバ計算機５０１も保持していると考えられる。これは、中継経路上にある２次キャッシュｐｒｏｘｙサーバ計算機５０１と１次キャッシュｐｒｏｘｙサーバ計算５０３～５０８では必ずファイルオブジェクトがキャッシュされるからである。また、各１次キャッシュｐｒｏｘｙサーバ計算機同士でも、別の１次キャッシュｐｒｏｘｙサーバ計算機に接続されるクライアント計算機が同一のファイルオブジェクトを読出している場合、同一のファイルオブジェクトを重複して持っている可能性が高い。

【００６６】したがって、分散ファイルシステム全体でのキャッシュ容量は２次キャッシュｐｒｏｘｙサーバ計算機５０１のキャッシュファイル５０２の容量であるといえるので、１台の２次キャッシュｐｒｏｘｙサーバ計算機だけではキャッシュファイルの容量には限界があるといえる。

【００６７】このような問題を解決するために、図２４のような方式が提案され実施されている。

【００６８】この方式では、図２３の２次キャッシュｐｒｏｘｙサーバ計算機５０１に相当するｐｒｏｘｙサーバ計算機は存在しない。各ｐｒｏｘｙサーバ計算機６０１～６０６は、クライアント計算機群を分担して受け持っている。たとえば、ｐｒｏｘｙサーバ計算機６０２に接続されたクライアント計算機２４からの読出要求は、ｐｒｏｘｙサーバ計算機６０２に受信される。ここで、ｐｒｏｘｙサーバ計算機６０２のキャッシュファイル６０８に有効なファイルオブジェクトが存在する場合、当該ファイルオブジェクトを取出し、読出要求を出したクライアント計算機２４に送出する。

【００６９】ｐｒｏｘｙサーバ計算機６０２のキャッシュファイル６０８に有効なファイルオブジェクトが存在しなかった場合、ｐｒｏｘｙサーバ計算機６０２はローカルネットワーク６００に接続される他のｐｒｏｘｙサーバ計算機６０１，６０３～６０６に対し、それぞれに接続されるキャッシュファイル６０７，６０９～６１２内に有効なファイルオブジェクトを保持しているかどうかを問合せる。有効なファイルオブジェクトを保持しているｐｒｏｘｙサーバ計算機が存在する場合には、たとえば、ｐｒｏｘｙサーバ計算機６０５がキャッシュファイル６１１に有効なファイルオブジェクトを保持している場合、ｐｒｏｘｙサーバ計算機６０５はｐｒｏｘｙサーバ計算機６０２に対してファイルオブジェクトを送信し、ｐｒｏｘｙサーバ計算機６０２は当該ファイルオブジェクトを読出要求を出したクライアント計算機２４に送出する。

【００７０】ローカルネットワーク６００に接続されたｐｒｏｘｙサーバ計算機６０１，６０３～６０６のキャッシュファイル６０７，６０９～６１２に有効なファイルオブジェクトが存在しなかった場合、ｐｒｏｘｙサーバ計算機６０２は広域ネットワーク６１を経由して、サーバ計算機１１にクライアント計算機２４からの読出要求を中継する。

【００７１】以降の処理は、単体のｐｒｏｘｙサーバ計算機と同様であるので、ここでの詳細な説明は繰返さない。

【００７２】この方式により、同一のファイルオブジェクトを複数のｐｒｏｘｙサーバ計算機が保持することがなくなり、キャッシュファイルの有効利用が図れる。また、分散ファイルシステム全体でのキャッシュ容量は、全ｐｒｏｘｙサーバ計算機６０１～６０６のキャッシュファイル６０７～６１２の容量の合計となり、キャッシュの大容量化が図れる。したがって、高いキャッシュヒット率が期待でき、実行転送速度の向上が期待できる。

【００７３】また、図２５のような方式も提案され実施されている。図２３の方式との違いは、２次キャッシュｐｒｏｘｙサーバ計算機７０５～７０７が複数台用意され、１次キャッシュｐｒｏｘｙサーバ計算機７１１～７１６が２次キャッシュｐｒｏｘｙサーバ計算機７０５～７０７にクライアント計算機群からの読出要求を中継する場合、その読出要求の要求先であるサーバ計算機７０

１，７０２のアドレス名称によって、たとえば、名称がｘｘｘ．ｅｄｕならばｅｄｕドメインサーバ計算機７０２に対する読出要求と判断して、ｐｒｏｘｙサーバ計算機７０５にアクセス要求を出力する。また、名称がｙｙｙ．ｃｏｍならばｃｏｍドメインサーバ計算機７０２に対する読出要求と判断してｐｒｏｘｙサーバ計算機７０６にアクセス要求を出力する。このように、末尾のドメイン名称（ｅｄｕ，ｃｏｍ等）による簡単なルールでアクセス要求を中継する２次キャッシュｐｒｏｘｙサーバ計算機７０５～７０７を選択している。

【００７４】その他の処理は単一のｐｒｏｘｙサーバ計算機と同一であるので、ここでの詳細な説明は繰返さない。

【００７５】これにより、同一の２次キャッシュｐｒｏｘｙサーバ計算機に読出要求が集中することがなくなり、負荷の分散が図れる。この方式においても、同一のファイルオブジェクトを複数の２次キャッシュｐｒｏｘｙサーバ計算機が保持する必要はなく、キャッシュファイルの有効利用が図れる。また、分散ファイルシステム全体でのキャッシュ容量は全２次キャッシュｐｒｏｘｙサーバ計算機のキャッシュファイルの容量の合計となり、キャッシュの大容量化が図れる。したがって、高いキャッシュヒット率が期待でき、実行転送速度の向上が期待できる。

【００７６】

【発明が解決しようとする課題】上述したｐｒｏｘｙサーバ装置においては、キャッシュ有効期限をある程度の長さに設定することによりキャッシュファイルに蓄積されるファイルオブジェクトの容量が増え、キャッシュヒット率が高まり効率的なファイルオブジェクトの転送が可能になる。

【００７７】しかし、一方、サーバ計算機のファイルオブジェクトの内容は、修正変更されることがあるため、キャッシュファイルの有効期限が長いと、その期間内はサーバ計算機内のファイルオブジェクトの内容が変更されているにもかかわらず、内部ネットワークのクライアント計算機のユーザはキャッシュファイルに蓄積された古い内容のファイルオブジェクトを読出すことになる。

【００７８】たとえば、天気図や電子新聞等のファイルオブジェクトは、同一のファイルオブジェクト名称でありながら数時間おきに情報更新されるものが多い。したがって、たとえば、キャッシュ有効期限を２４時間とすると、３時間おきに更新される天気図、電子新聞などの情報は７／８以上の確率で古くて役に立たない情報となってしまう。

【００７９】図２６は、このような従来のｐｒｏｘｙサーバ装置におけるキャッシュ制御のデータの流れをタイムチャートにしたものである。Ａ，Ｂ，Ｃ，Ｄ，Ｅ等の名称は同一ファイルオブジェクト名称でありながら内容が異なるデータであることを示している。クライアント計算機がｐｒｏｘｙサーバ装置経由でアクセス（ユーザアクセス（１））すると、サーバ計算機のファイルオブジェクトの内容Ｂがｐｒｏｘｙサーバ計算機によってキャッシュされ、クライアント計算機にはＢが転送される。次に、クライアント計算機からユーザアクセス（２）が送信されると、ｐｒｏｘｙサーバ装置にキャッシュされている内容Ｂは２４時間有効なので、キャッシュヒットしてファイルオブジェクトの内容Ｂがクライアント計算機に転送される。しかしこのときには、既にサーバ計算機内では内容がＥとなっているのに、クライアント計算機はＢという内容の古いデータを見ていることになる。

【００８０】このようなキャッシュファイルの内容のうち、外部ネットワークに接続されたサーバ計算機内に存在する元のデータが変更されている割合をステールデータ率（stale date rate）と呼ぶ。したがって、キャッシュ有効期限を増やすとキャッシュファイルに蓄積されるファイルオブジェクトの容量が増加してキャッシュヒット率が高くなるが、キャッシュファイル内の内容のステールデータ率が増加することになる。

【００８１】上述したＣＥＲＮのｈｔｔｐｄソフトウェアで構成されるｐｒｏｘｙサーバ装置であれば、特定のサーバ計算機のファイルオブジェクトのキャッシュ有効期限を小さくできるが、この方法ではクライアント計算機がファイルオブジェクトにアクセスを行なうと、キャッシュ有効期限が短いためキャッシュにミスヒットしてからサーバ計算機にファイルオブジェクトに対するアクセスを行なうことになる。したがって、クライアント計算機のユーザは、ｐｒｏｘｙサーバ計算機によるキャッシュの恩恵を受けられず、広域ネットワークによる低速なファイルオブジェクト転送が終了するまで待たされることになる。

【００８２】したがって上述した従来の技術では、キャッシュファイルの中のファイルオブジェクトの内容の新鮮さとキャッシュヒット率の維持はそれぞれ相反する問題となる。

【００８３】一般に、キャッシュファイルのステールデータ率が増加すると、クライアント計算機のユーザは内容が古いはずであると自主的に判断してファイルオブジェクトをサーバ計算機から直接ロードするようになる。この場合キャッシュファイルを通り越して、実際にサーバ計算機からロードすることを指定するプロトコルが利用される。その結果、ｐｒｏｘｙサーバ計算機のキャッシュファイルが意味を持たなくなることになる。

【００８４】また、従来よりｐｒｏｘｙサーバ計算機の負荷を分散する方法として、上述したように図２３～図２５の方式が提案されている。しかしいずれの場合にも十分に負荷が分散されているとはいえない。図２３に示す方式では、１台の２次キャッシュｐｒｏｘｙサーバ計算機５０１が広域ネットワークに接続されているサーバ

計算機との中継をすべて行なっているので、負荷の分散が達成されているとはいえない。

【００８５】図２４の方式では、ｐｒｏｘｙサーバ計算機６０１～６０６と広域ネットワーク６１との間の経路が多重化されていない。また、ローカルネットワーク６００に接続されるｐｒｏｘｙサーバ計算機６０１～６０６は、この方式に対応した特殊なプロトコルを理解するものに置き換える必要がある。

【００８６】図２５の方式では、１次キャッシュｐｒｏｘｙサーバ計算機７１１～７１６は、ドメイン名等の単純なルールで分散を行なうので、２次キャッシュｐｒｏｘｙサーバ計算機７０５～７０７間での負荷の偏りが生じる。

【００８７】本発明は、上記問題点を解決するためになされたもので、請求項１に記載の発明の目的は、キャッシュファイルの内容を新鮮に保つとともに、キャッシュヒット率も向上させることである。

【００８８】請求項２に記載の発明の目的は、キャッシュファイルの内容のうちサーバ計算機における更新頻度の高いファイルオブジェクトを新鮮に保つとともに、キャッシュヒット率も向上させることである。

【００８９】請求項３に記載の発明の目的は、通常のユーザの利用に悪影響を与えることなく、キャッシュファイルの内容を新鮮に保つとともに、キャッシュヒット率も向上させることである。

【００９０】請求項４から６に記載の発明の目的は、キャッシュファイルの内容を新鮮に保つとともに、キャッシュヒット率も向上させることである。

【００９１】請求項７または８に記載の発明の目的は、ファイルオブジェクトに対するアクセス要求を分散して経路を有効に利用し、スループットを向上させることである。

【００９２】請求項９から１１に記載の発明の目的は、ファイルオブジェクトに対するアクセス要求を分散して経路を有効に利用し、スループットを向上させ、さらにローカルネットワーク上の通信量を削減することである。

【００９３】請求項１２に記載の発明の目的は、ファイルオブジェクトに対するアクセス要求を分散して経路を有効に利用し、スループットを向上させることである。

【００９４】

【課題を解決するための手段】請求項１に記載の発明は、クライアント計算機が接続される第１のネットワークと、サーバ計算機が接続される第２のネットワークとの間に介在するように設けられるゲートウェイ装置であって、第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトにより指定される第２のネットワークのサーバ計算機に対して、当該ファイルオブジェクトに対するアクセス要求を行なうネットワーク中継手段を含み、ネットワークファイル中継手段は、取得したファイルオブジェクトを所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、過去の一定期間内に行なわれたファイルオブジェクトの転送記録を蓄積するファイル転送記録手段と、ファイルオブジェクトに対するアクセス要求に応答して当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在する場合は当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在しない場合には当該ファイルオブジェクトにより指定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含み、ゲートウェイ装置はさらに、ファイル転送記録手段から所定の規則に従って転送記録を抽出し、当該転送記録のファイルオブジェクトを第２のネットワークのサーバ計算機から取得してキャッシュファイル手段に格納するためのキャッシュ更新手段を含む。

【００９５】ネットワークファイル中継手段によって転送されたファイルオブジェクトは、所定の規則に従ってキャッシュ更新手段により更新される。キャッシュ更新手段は、有効期限が切れたものはもちろん、有効期限内のファイルオブジェクトも第２のネットワークのサーバ計算機から取得してキャッシュファイル手段に格納する。そのため、キャッシュの有効期限を長めに取っておいても、キャッシュの有効期限よりも短い間隔で変更されるサーバ計算機のファイルオブジェクトがキャッシュに反映される確率は高くなる。

【００９６】請求項２に記載のゲートウェイ装置は、請求項１に記載のゲートウェイ装置であって、キャッシュ更新手段はさらに特定の文字列パターンを予め記憶するための文字列パターン記憶手段を含み、キャッシュ更新手段は特定の文字列パターンと一致するファイルオブジェクト名称を含む転送記録をファイル転送記録手段から抽出し、当該ファイルオブジェクトを第２のネットワークのサーバ計算機から取得してキャッシュファイル手段に格納する。

【００９７】キャッシュ更新手段は、特定の文字列パターンと一致するファイルオブジェクトのみ更新アクセスを行なうので、予めファイルオブジェクトの更新頻度の高いファイルオブジェクト名称を登録しておけば不要なキャッシュ更新アクセスを削減でき、キャッシュファイルの内容を新鮮に保つとともにキャッシュヒット率も向上させることができる。

【００９８】請求項３に記載のゲートウェイは、請求項１に記載のゲートウェイ装置であって、ネットワークファイル中継手段は、各々が第１のネットワークのクライアント計算機からのファイルオブジェクトのアクセス要

求を処理するための第１の転送プロセスを複数個、同時並列的に起動させ、キャッシュ更新手段は各々が１つのファイルオブジェクトの取得を行なうための第２の転送プロセスを複数個、同時並列的に起動させ、かつ第１の転送プロセスの数を検知して、第１の転送プロセスの数と第２の転送プロセスの数との和が予め定められる上限以下となるように第２の転送プロセスの新たな起動を制御する。

【００９９】第１のプロセスが多数発生しているときは第２のプロセスの起動を抑制し、第１のプロセスが僅かしか発生していないときには第２のプロセスを多数起動させることができるので、通常のユーザによるファイルオブジェクトのアクセス要求が、キャッシュ更新手段による悪影響を受けることがない。また、通常のユーザによるアクセス要求が少ないときには随時更新処理が行なえるので、キャッシュの新鮮さを保つことができる。

【０１００】請求項４に記載のゲートウェイ装置は、請求項１に記載のゲートウェイ装置であって、キャッシュ更新手段はファイル転送記録手段の中から、キャッシュファイル手段からキャッシュヒットしたファイルオブジェクトの転送記録を抽出し、当該ファイルオブジェクトを第２のネットワークのサーバ計算機から取得してキャッシュファイル手段に格納する。

【０１０１】ユーザによるアクセス要求がキャッシュにヒットした場合、、キャッシュ更新手段はその直後にサーバ計算機へファイルオブジェクトのアクセス要求を行なう。したがって別ユーザが同一ファイルオブジェクトをアクセスした場合、最新のファイルオブジェクトが既にキャッシュに格納されていることになる。

【０１０２】請求項５に記載のゲートウェイ装置は、請求項１に記載のゲートウェイ装置であって、キャッシュ更新手段はさらにファイル転送記録手段の末尾から転送記録を順次読出して格納し、先に格納された転送記録から順に出力するＦＩＦＯバッファ手段を含み、キャッシュ更新手段はＦＩＦＯバッファ手段から転送記録を読出して、当該転送記録のファイルオブジェクトを第２のネットワークのサーバ計算機から取得してキャッシュファイル手段に格納する。

【０１０３】ＦＩＦＯバッファ手段は、ファイル転送記録手段の末尾から転送記録を順次格納し出力するので、キャッシュ更新手段の処理が容易になる。

【０１０４】請求項６に記載のゲートウェイ装置は、請求項１に記載のゲートウェイ装置であって、キャッシュ更新手段はファイル転送記録手段から抽出した転送記録のファイルオブジェクトの前回の更新が所定時間以内であれば、第２のネットワークのサーバ計算機へのアクセスは行なわない。

【０１０５】キャッシュ更新手段は、所定時間以内に再び同じファイルオブジェクトに対するアクセス要求があった場合に後のアクセス要求に対するキャッシュ更新処理を行なわない。このことによって、不要なキャッシュ更新アクセスを削除し、無駄なトラフィックを抑制して回線を有効に使うことができる。

【０１０６】請求項７に記載の発明は、クライアント計算機と第１のネットワークによって接続され、かつサーバ計算機と第２のネットワークによって接続される複数のキャッシュサーバ計算機と第３のネットワークによって接続されるゲートウェイ装置であって、第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行なうための演算手段と、当該演算結果に基づいて第３のネットワークの複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択するための中継制御手段とを含む。

【０１０７】中継制御手段は、複数のキャッシュサーバ計算機の中からファイルオブジェクトに対するアクセス要求を出力するキャッシュサーバ計算機を選択する場合に、演算結果が一意的に定まる演算を行なうことによってキャッシュサーバ計算機を一意的に定める。したがって同一のファイルオブジェクトに対するアクセス要求は同一の経路によって転送され、異なるファイルオブジェクトに対するアクセス要求は分散して経路を転送することになるので、経路が有効に利用され、スループットが向上することになる。

【０１０８】請求項８に記載の発明は、クライアント計算機とゲートウェイ装置とが第１のネットワークで接続され、サーバ計算機とキャッシュサーバ計算機とが第２のネットワークで接続され、複数のゲートウェイ装置と複数のキャッシュサーバ計算機とが第３のネットワークで接続される分散ファイルシステムであって、ゲートウェイ装置は、第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行なうための演算手段と、当該演算結果に基づいて第３のネットワークの複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択するための中継制御手段とを含む。

【０１０９】各ゲートウェイ装置は、アクセス要求に含まれる所定のデータに同一の演算を行なうため複数のキャッシュサーバ計算機の中からキャッシュサーバ計算機を選択する際、同一のファイルオブジェクトであれば同一のキャッシュサーバ計算機を選択することになる。したがって同一のファイルオブジェクトが複数のキャッシュサーバ計算機内に含まれるということがなくなり、キャッシュの有効利用が可能となる。

【０１１０】請求項９に記載の発明は、クライアント計算機が接続される第１のネットワークと、サーバ計算機が接続される第２のネットワークとの間に介在するよう

に設けられるゲートウェイ装置であって、第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、この演算結果に基づいてゲートウェイ装置を選択し、この選択が他のゲートウェイ装置であれば、アクセス要求を第３のネットワークを介して他のゲートウェイ装置へ出力するための中継制御手段と、第２のネットワークのサーバ計算機から取得したファイルオブジェクトを所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、過去の一定時間内に行われたファイルオブジェクトの転送記録を蓄積するためのファイル転送記録手段と、中継制御手段が自身のゲートウェイ装置を選択した場合と第３のネットワークを介して他のゲートウェイ装置からアクセス要求を受信した場合に、ファイルオブジェクトに対するアクセス要求に応答して、当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在する場合は当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在しない場合には当該ファイルオブジェクトにより所定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含む。

【０１１１】このゲートウェイ装置は、複数のキャッシュサーバ計算機の中からキャッシュサーバ計算機を選択する機能と、ファイルオブジェクトをキャッシュする機能とを有することによって、ローカルネットワーク上での通信量を削減することを可能にしている。

【０１１２】請求項１０に記載の発明は、サーバ計算機と複数のキャッシュサーバ計算機とが第２のネットワークで接続され、複数のキャッシュサーバ計算機と第１のネットワークによって接続されるクライアント計算機であって、キャッシュサーバ計算機に対するファイルオブジェクトのアクセス要求を出力し、当該ファイルオブジェクトを取得するためのファイル取得手段と、キャッシュサーバ計算機へのファイルオブジェクトに対するアクセス要求を出力する際、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、当該演算結果に基づいて複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択するための中継制御手段とを含む。

【０１１３】クライアント計算機に、複数のキャッシュサーバ計算機の中からキャッシュサーバ計算機を選択する機能を含ませることによって、ローカルネットワーク上での通信量を削減し、クライアント計算機の有効利用を図っている。

【０１１４】請求項１１に記載のクライアント計算機

は、請求項１０に記載のクライアント計算機であって、中継制御手段はさらに、第３のネットワークを介して他のクライアント計算機からのファイルオブジェクトに対するアクセス要求を受信するための受信手段を含み、中継制御手段は他のクライアント計算機からのアクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、当該演算結果に基づいて複数のキャッシュサーバ計算機の中からアクセス要求を出力するキャッシュサーバ計算機を選択する。

【０１１５】このクライアント計算機は、他のクライアント計算機からのアクセス要求を受信して、当該アクセス要求に含まれる所定のデータに基づいて複数のキャッシュサーバ計算機の中からキャッシュサーバ計算機を選択している。したがって同じローカルネットワークに接続されるクライアント計算機の中の一部にキャッシュサーバ計算機を選択する機能を持たせて、クライアント計算機の有効利用を図っている。

【０１１６】請求項１２に記載の発明は、クライアント計算機と第１のネットワークによって接続され、かつサーバ計算機と第２のネットワークによって接続される複数のキャッシュサーバ計算機と第３のネットワークによって接続されるゲートウェイ装置であって、第１のネットワークのクライアント計算機からのファイルオブジェクトに対するアクセス要求に対して、当該アクセス要求に含まれる所定のデータに演算結果が一意的に定まる所定の演算を行ない、当該演算結果に基づいてゲートウェイ装置を選択し、この選択が他のゲートウェイ装置であれば、当該アクセス要求を第３のネットワークを介して他のゲートウェイ装置へ出力するための中継制御手段と、第２のネットワークのサーバ計算機から取得したファイルオブジェクトを、所定の有効期限が経過するまで一時的に蓄積するとともに、最終変更時刻をファイルオブジェクト単位で記録するためのキャッシュファイル手段と、過去の一定時間内に行なわれたファイルオブジェクトの転送記録を蓄積するためのファイル転送記録手段と、中継制御手段が自身のゲートウェイ装置を選択した場合と第３のネットワークを介して他のゲートウェイ装置からアクセス要求を受信した場合に、ファイルオブジェクトに対するアクセス要求に応答して当該ファイルオブジェクトの最終変更時刻を参照し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在する場合には当該ファイルオブジェクトをクライアント計算機に転送し、有効期限内のファイルオブジェクトがキャッシュファイル手段に存在しない場合には当該ファイルオブジェクトにより指定されるサーバ計算機の当該ファイルオブジェクトを取得してクライアント計算機に転送するための転送手段とを含む。

【０１１７】ゲートウェイ装置に、キャッシュサーバ計算機を選択する機能と、ファイルオブジェクトをキャッシュする機能とを持たせることによって、ローカルネッ

トワーク上の通信量を削減し、キャッシュサーバ計算機の負荷を軽減することを可能にしている。

【０１１８】

【発明の実施の形態】

［実施の形態１］本発明が適用されるゲートウェイ装置（ｐｒｏｘｙサーバ装置と呼ぶ）は、ネットワークインタフェースを備えたＵＮＩＸ　ＯＳの動作する計算機によって、以下の説明および図面を参照すれば容易に実現可能である。そうしたｐｒｏｘｙサーバ装置の構成の一例は、図１９に示したとおりである。図１９のｐｒｏｘｙサーバ装置については既に説明したので、その構成についての詳細な説明はここでは繰返さない。

【０１１９】通常は、ユーザ３１がクライアント計算機２４を使用してｐｒｏｘｙサーバ装置（図１９におけるｐｒｏｘｙサーバ装置４００）に対して、サーバ計算機１１のファイルオブジェクト１０にを取得するようネットワーク２３を経由して依頼を出す。ファイルオブジェクト１２は、前述したようにプロトコル名称と、サーバ計算機のネットワークアドレスと、ファイルオブジェクトの名称と、ファイルデータの実体との組からなっている。

【０１２０】図１に本願発明に係るｐｒｏｘｙサーバ装置の第１の実施の形態における装置５３のシステム構成を示す。ｐｒｏｘｙサーバ装置５３は、内部ネットワーク２３および外部ネットワーク２５に接続された第１のｐｒｏｘｙプロセス１４と、第１のｐｒｏｘｙプロセス１４が使用するキャッシュファイル１６と、第１のｐｒｏｘｙプロセス１４が、アクセスログ（転送記録）としてファイルの名称を記録するアクセスログ１５と、アクセスログ１５の末尾から順次取出して格納し、先に格納されたアクセスログから順に出力するＦＩＦＯバッファ３５と、予め作成されたアクセスパターンリストを格納するパターンファイル３０と、ＦＩＦＯバッファ３５からアクセスログを取出してパターンファイル３０に格納されたアクセスパターンと比較し一致する場合にはこのアクセスログをメモリバッファＵＲＬＢＵＦ（ＵＲＬについては後述する）に格納するキャッシュ更新プロセス３６と、ＵＲＬＢＵＦに格納されたアクセスログに含まれるファイルオブジェクト名称をもとに、当該ファイルオブジェクトをサーバ計算機１１から取得してキャッシュファイル１６に格納する第２のｐｒｏｘｙプロセス３４とを含む。

【０１２１】第１のｐｒｏｘｙプロセス１４と、第２のｐｒｏｘｙプロセス３４と、キャッシュ更新プロセス３６とはいずれも、たとえば図１９に示すメモリ２０６内にプログラムとして格納され、ＣＰＵ２０２によってこれを実行することにより実現される。

【０１２２】第１のｐｒｏｘｙプロセス１４が、特許請求の範囲に記載の転送手段に相当し、キャッシュファイル１６がキャッシュファイル手段に相当し、アクセスログ１５がファイル転送記録手段に相当する。また、キャッシュ更新プロセス３６と、パターンファイル３０と、ＦＩＦＯバッファ３５と、第２のｐｒｏｘｙプロセス３４とによりキャッシュ更新手段が構成される。

【０１２３】第２のｐｒｏｘｙプロセス３４は、本願発明に係るゲートウェイ装置に特有なものであるが、次のような条件に従って起動される。

【０１２４】（１）　第２のｐｒｏｘｙプロセス３４は、第１のｐｒｏｘｙプロセス１４と共通のキャッシュファイル１６を使用する。

【０１２５】（２）　第２のｐｒｏｘｙプロセス３４のキャッシュ有効期限は０に設定される。すなわち、第２のｐｒｏｘｙプロセスを経由するサーバ計算機１１へのアクセスは、必ず外部ネットワーク２５内のサーバ計算機１１に対して行なわれ、ファイルオブジェクトを取得してキャッシュファイル１６に格納する。

【０１２６】図１に示すｐｒｏｘｙサーバ装置５３の動作について以下説明する。内部ネットワーク２３内のクライアント計算機２４が、このｐｒｏｘｙサーバ装置５３に対してネットワークアクセス要求１９を与えるものとする。第１のｐｒｏｘｙプロセス１４が、このネットワークアクセス要求１９を受け取り、まずＩ／Ｏインタフェース２０２（図１７、図１８、図１９参照）を経由してファイル装置２１６内に存在するキャッシュファイル１６をアクセスする（２１）。また第１のｐｒｏｘｙプロセス１４は、このネットワークアクセス要求とキャッシュファイル１６内のファイルオブジェクト名称とを比較し（２２）、一致しているか否かを判定する。一致したものがあれば第１のｐｒｏｘｙプロセス１４はさらに、当該ファイルオブジェクトのキャッシュファイル１６内の最終変更時刻を抽出し（２２）、現在の時刻と比較する。そして現在の時刻が、当該ファイルオブジェクトの最終変更時刻を起点とする有効期限内であれば、キャッシュファイル１６から当該ファイルオブジェクトを抽出し（２２）、ネットワークＩ／Ｏインタフェース４０２を経由してクライアント計算機２４にファイルオブジェクトを返す（２０）。

【０１２７】該当するファイルオブジェクトがキャッシュファイル１６内に存在していない場合、または存在していてもそのキャッシュ有効期限が切れている場合には、第１のｐｒｏｘｙプロセス１４は、ネットワークＩ／Ｏインタフェース４０２を経由して外部ネットワーク２５のサーバ計算機１１にファイルオブジェクト転送要求１７を送る。このときのサーバ計算機１１は、当該ファイルオブジェクト内のネットワークアドレス名称により特定される。

【０１２８】該当するサーバ計算機１１は、要求されたファイルオブジェクトを第１のｐｒｏｘｙプロセス１４に返送する（１８）。すなわちこのファイルオブジェクトが第１のｐｒｏｘｙプロセス１４により取得される。

【０１２９】第１のｐｒｏｘｙプロセス１４は、このファイルオブジェクトをキャッシュファイル１６にその最終変更時刻とともに書込み（２１）、アクセスログ１５にこのファイルオブジェクトの名称を記録する（２６）。

【０１３０】一方、本願発明のｐｒｏｘｙサーバ装置５３は、キャッシュ更新プロセス３６と第２のｐｒｏｘｙプロセス３４とをＣＰＵ２０２により処理走行させてファイルオブジェクトを更新する点において従来のｐｒｏｘｙサーバ装置と相違する。

【０１３１】キャッシュ更新プロセス３６および第２のｐｒｏｘｙプロセス３４は次のように動作する。第１のｐｒｏｘｙプロセス１４によってアクセスログファイル１５に格納されたアクセスログの末尾からＦＩＦＯバッファ３５に順に格納される（２８）。ＦＩＦＯバッファ３５は、先に格納されたアクセスログから順に出力する（２７）。キャッシュ更新プロセス３６は、ＦＩＦＯバッファ３５からアクセスログを１行ずつ抽出し、メモリバッファに格納する。キャッシュ更新プロセス３６は、メモリバッファからアクセスログを取出し、それぞれのアクセスログをもとにキャッシュにヒットしているか否かを判定する。キャッシュにヒットしていればさらに、パターンファイル３０からアクセスパターンリストを抽出し（２９）、アクセスログに含まれるファイルオブジェクト名称がアクセスパターンリストと一致しているものを抽出する。またキャッシュ更新プロセス３６は、抽出したファイルオブジェクトのうちファイルオブジェクト名称が特定のパターンに一致するものは除外して、これをメモリバッファＵＲＬＢＵＦに格納する。

【０１３２】第２のｐｒｏｘｙプロセス３４は、ＵＲＬＢＵＦが空でないならば、ＵＲＬＢＵＦに格納されたファイルオブジェクト名称をもとに（３１）、サーバ計算機１１にアクセスし（３２）、当該ファイルオブジェクトを取得して（３３）キャッシュファイル１６に格納する（３７）。これによりキャッシュファイル１６内のファイルオブジェクトが最新の内容に維持される。

【０１３３】このように、第２のｐｒｏｘｙプロセス３４を起動すると、大きく次の２つの効果を得ることができる。

【０１３４】（１）　第１のｐｒｏｘｙプロセス１４では、キャッシュファイルの有効期限がある値に設定されており、ファイルオブジェクトのキャッシュ更新処理をこの第１のｐｒｏｘｙプロセス１４を用いて行なったとすると、キャッシュファイル１６にヒットしてしまう可能性が高い。これでは、外部ネットワーク２５のサーバ計算機１１のファイルオブジェクトに対するアクセスが行なわれないので、キャッシュファイル１６の更新が実現できない。

【０１３５】一方、第２のｐｒｏｘｙプロセス３６は、キャッシュ有効期限が０になっており、第２のｐｒｏｘｙプロセス３６に対してアクセス要求を出せば、このキャッシュファイルはミスヒットしたことになるので、外部ネットワーク２５を介してサーバ計算機１１に対するファイルオブジェクトのアクセスが行なわれ（３２）、当該ファイルオブジェクトは第２のｐｒｏｘｙプロセス３４に転送される（３３）ことによって、キャッシュファイル１６中のファイルオブジェクトは最新の状態に更新される。

【０１３６】（２）　第１のｐｒｏｘｙプロセス１４は、アクセスログを残すが、このアクセスログをもとにキャッシュ更新アクセスをするべきファイルオブジェクトが抽出される。したがって、もしキャッシュ更新プロセス３６が第１のｐｒｏｘｙプロセス１４を使用してサーバ計算機１１のファイルオブジェクトのアクセスを行なうと、その結果として生成されるアクセスログがアクセスファイル１５に記録されてしまう。キャッシュ更新プロセス３６は、次回の更新アクセス動作においてアクセスログファイル１５に格納されたアクセスログをもとにアクセスすべきファイルオブジェクト名称を抽出する。したがって、再度同じファイルオブジェクトをサーバ計算機１１から取得することになり、同一のファイルオブジェクトが多重に出現するループ動作が起きてしまうことになる。これを避けるために第２のｐｒｏｘｙプロセス３４を動作させ、第２のｐｒｏｘｙプロセス３４を利用したキャッシュ更新アクセス動作を行なう。

【０１３７】本発明を非常に有効に適用できる例として、インターネット上のWWWシステムがある。WWWシステム上でのｐｒｏｘｙサーバ装置において、本願発明のキャッシュ更新プロセスを行なう手順を以下に説明する。

【０１３８】WWWシステムでは、ネットワーク上に分散したファイルオブジェクト名称はUniform Resource Locator（ＵＲＬ）と呼ばれる形式で表現され、特定される。ＵＲＬの一例を次に示す。

【０１３９】

【数４】

http://www.xxx.co.jp/test/index.html

【０１４０】数４においては「ｈｔｔｐ」は使用するプロトコルを示す。「ｗｗｗ．ｘｘｘ．ｃｏ．ｊｐ」は、ネットワーク上のＨＴＴＰサーバ計算機のアドレスを示すものであり、ネットワーク上で同一のものはないよう選ばれている。また「／ｔｅｓｔ／ｉｎｄｅｘ．ｈｔｍｌ」はサーバ計算機内のファイル名称を示す。

【０１４１】第１のｐｒｏｘｙプロセス１４として一般に利用されているＤｅｌｅＧａｔｅを使用する場合は、そのアクセスログの例は次のように、クライアント計算機名称、時刻、“ＨＴＴＰプロトコル（ファイル取得要求）”、その他、およびキャッシュヒット状況（キャッシュヒットした場合H）などとなる。以下に、そのアクセスログの例を示す。

29 30

【０１４２】 ＊ ＊【数５】

```
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/sharpcolor.gif HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/isg.gif HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/Zaurus.gif HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/Shoin.gif HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/Prostation.gif HTTP/1.0"その他H
ｸﾗｲｱﾝﾄ計算機名称 [時刻] "GET  http://naragw.sharp.co.jp/S2.gif HTTP/1.0"その他H
```

【０１４３】数５に示すアクセスログは、図２に示すような、出願人がインターネット上で公開している情報ページへアクセスした場合のアクセスログに相当する。図２に示されるページは、テキストと、６個のグラフィックデータとから構成されているので、このページに対するアクセスを１回行なうと、６個のグラフィックデータに対するアクセスを含む７つのアクセスログが形成される。すなわち、上述のアクセスログにおいて第１行目はこの情報ページのテキストに対するアクセス要求であり、残りの６行は表紙に貼り込まれた出願人会社のロゴのグラフィックデータ（ｓｈａｒｐｃｏｌｏｒ．ｇｉ ※ｆ）、出願人会社内の情報システム事業本部と呼ばれる事業部のロゴ（ｉｓｇ．ｇｉｆ）、出願人会社の４つの製品の写真のグラフィックデータ（Ｚａｕｒｕｓ．ｇｉｆ，Ｓｈｏｉｎ．ｇｉｆ，Ｐｒｏｓｔａｔｉｏｎ．ｇｉｆ，Ｓ２．ｇｉｆ）の取得要求を示している。

【０１４４】ＨＴＴＰプロトコルは、コマンド文字列（ＧＥＴ等）と、ＵＲＬと、プロトコルバージョン（「ＨＴＴＰ／１．０」等）から構成されている。以下はその一例である。

【０１４５】

【数６】

```
"GET  http://www.xxx.co.jp/test/index.html  HTTP/1.0"
```

【０１４６】ＨＴＴＰプロトコルについては、最後に掲げる表内に示された参考文献のうち、参考文献３および参考文献４に記載されている。キャッシュ更新プロセスの実施手順について以下に説明するが、そのための前提条件は次の（１）および（２）となっている。

【０１４７】（１） 第１のｐｒｏｘｙプロセス１４は、キャッシュ有効期限Ｍ（時間）で起動する。たとえばＭ＝２４時間とする。

【０１４８】（２） 第２のｐｒｏｘｙプロセス３４は、①キャッシュファイルを第１のｐｒｏｘｙプロセス１４のものと共有し、②キャッシュ有効期限を０に設定する、という２つの条件に従って起動される。

【０１４９】このような条件の下に、本願発明のキャッシュ更新プロセスを毎日一定時刻に起動し、一定時刻に終了する。毎日一定時刻にキャッシュ更新プロセスを自動起動するには、ＵＮＩＸ ＯＳに備わっているタイマ機能（ｃｒｏｎ）を使用する。これは、ＵＮＩＸ ＯＳに関連してよく知られている機能であるので、その詳細についてはここでは説明しない。なお、ＵＮＩＸ ＯＳ以外のＯＳでも、同様の機能が提供されていることが多い。

【０１５０】図３は、本願発明のキャッシュ更新プロセスの処理手順を示すフローチャートである。このプロセス☆スを実現するためのプログラムは、たとえば図１９のメモリ２０６内に配置され、ＣＰＵ２０２によって実行される。このプロセスとネットワークおよびファイル装置２１６との入出力には、それぞれネットワークＩ／Ｏインタフェース４０２とＩ／Ｏインタフェース２０８とが使用される。

【０１５１】まず、ステップＳ１において第１のｐｒｏｘｙプロセス１４が、アクセスログファイル１５に格納されるアクセスログの末尾を１行読出す。読出せない場合は、アクセスログが読めるまで待機する。ＵＮＩＸでは、このようなアクセスログの末尾を呼出す機能は、ＯＳとともに提供されているコマンドｔａｉｌで可能であり、その標準出力から１行読出し、メモリ内に確保された文字列バッファであるＬＩＮＥバッファに１行格納するには、以下のように行なう。ここで、第１のｐｒｏｘｙプロセス１４が格納したアクセスログが／ｕｓｒ／ｌｏｃａｌ／ｅｔｃ／ｔｅｌｅｇａｔｅｄ／ｌｏｇｓ／１０００１．ｈｔｔｐであるとする。これは図１のアクセスログファイル１５に格納されるアクセスログに相当する。

【０１５２】

【数７】

```
tail  -f  /usr/local/etc/delegated/logs/10001.http  |  read  LINE
```

【０１５３】数７のｔａｉｌ －ｆによる１行読出は、図１の２７に相当するが、これはソフトウェアによるＦＩＦＯバッファ３５を構成しており、読出ポインタおよび書込ポインタをそれぞれ有している。

【０１５４】

【数８】

31　　32

green [時刻] "GET http://naragw.sharp.co.jp/HTTP/1.0"その他H
green [時刻] "GET http://naragw.sharp.co.jp/sharpcolor.gifHTTP/1.0"その他H
rp－－＞green [時刻] "GET http://naragw.sharp.co.jp/isg.gifHTTP/1.0"その他H
green [時刻] "GET http://naragw.sharp.co.jp/Zaurus.gifHTTP/1.0"その他H
green [時刻] "GET http://naragw.sharp.co.jp/Shoin.gifHTTP/1.0"その他H
green [時刻] "GET http://naragw.sharp.co.jp/Prostation.gifHTTP/1.0"その他H
green [時刻] "GET http://naragw.sharp.co.jp/ZAURUS/index.htmlHTTP/1.0"その他H
wp－－＞　green [時刻] "GET http://naragw.sharp.co.jp/S2.gifHTTP/1.0"その他H

【０１５５】すなわち、第１のｐｒｏｘｙプロセス１４が書込むアクセスログは、１個のファイルオブジェクト転送ごとに１行発生するが、このとき書込ポインタ（ｗｐ）を１行進める。また、ソフトウェアによるＦＩＦＯバッファ３５がｔａｉｌ　－ｆでアクセスログを１行読出せば、読出ポインタ（ｒｐ）を１行進める。第１のｐｒｏｘｙプロセス１４による書込速度（行／秒）と、キャッシュ更新プロセス３６におけるｔａｉｌ　－ｆによる１行読出速度（行／秒）が一致していなくてもよい。第１のｐｒｏｘｙプロセス１４によるアクセスログ書込速度が大きくなると、書込ポインタ（ｗｐ）は先に進むが、キャッシュ更新プロセス３４によってこれを読出ポインタ（ｒｐ）が追いかけながらアクセスログが読出される。アクセスログ書込速度が小さいか、読出側のキャッシュ更新プロセス３４の処理が進行すれば、やがて読出ポインタ（ｒｐ）が書込ポインタ（ｗｐ）に追いつくことになる。これが、ソフトウェアＦＩＦＯバッファとしての動作である。

【０１５６】次に、ＬＩＮＥバッファに格納された１行のアクセスログ情報は過去にアクセスされたファイルオブジェクトの記録であるので、以下のような条件でキャッシュ更新アクセスを行なうべきファイルオブジェクト名称を含むものを選択する。この選択処理を、以後フィルタリングと呼び、図３に示すフローチャートのステップＳ２に相当する。このフィルタリングのルールは以下のとおりである。

【０１５７】（１）　ＨＴＴＰコマンド文字列がファイルオブジェクトに対するリード要求（ＧＥＴ）であるもの。

【０１５８】（２）　ファイルオブジェクト名称のプロトコル部分がｈｔｔｐであるもの。

（３）　キャッシュにヒットしたアクセスであるもの。

【０１５９】（４）　ファイルオブジェクト名称がキャッシュ更新アクセスに適さないものを除外する。

【０１６０】（５）　特定の文字列パターンを含むアクセスリストとファイルオブジェクト名称が一致するもの。

【０１６１】以上の（１）～（５）の条件を用いてフィルタリングを行ない、ファイルオブジェクト名称のフィールドだけを取出して、キャッシュ更新アクセスでファイルオブジェクト名称を抽出する。

【０１６２】ファイルオブジェクト名称のプロトコル部分がｈｔｔｐのもののみ抽出するのは、第１のｐｒｏｘｙプロセス１４および第２のｐｒｏｘｙプロセス３４として一般的に利用されているＤｅｌｅＧａｔｅソフトウェアがキャッシュファイルを生成するプロトコルはｈｔｔｐプロトコルのみであるので、そのファイルオブジェクトのみはキャッシュ更新アクセスする意味があるからである。それ以外のプロトコルによるアクセスはファイルオブジェクトがキャッシュされないので、キャッシュ更新アクセスをする意味がない。

【０１６３】キャッシュにヒットしたもののみを抽出するのは、キャッシュにヒットしていないファイルオブジェクトはサーバ計算機１１へのアクセスが行なわれたものと解釈されるので、その直後にキャッシュ更新アクセスを行なう必要はないと判断されるからである。

【０１６４】ファイルオブジェクト名称が、キャッシュ更新アクセスに適さないものとしては、ＵＲＬ中に「？」を含むものなどがある。「？」を含むＵＲＬ表記は、サーバ計算機１１に対してクライアント計算機２４を使用するユーザが文字列を人手で入力する場合などに使われるため、この文字を含むＵＲＬ表記を有するファイルオブジェクトをキャッシュ更新アクセスの対象とすると、予期せぬ結果を招くおそれがある。

【０１６５】特定の文字列パターンを含むアクセスリストと一致するもののみを選ぶのは、サーバ計算機において頻繁に内容更新が行なわれるものである天気図、電子新聞等を提供しているサーバ計算機など特定のものであることが多い。そこで、そのようなサーバ計算機のファイルオブジェクトを指定するような文字列パターンを格納したアクセスリストと一致したものだけをキャッシュ更新アクセスの対象とすることによって、不要なキャッシュ更新アクセスを極力削減するためのものである。

【０１６６】このような文字列パターンを格納したパターンファイル３０を、ここでｐａｔｔｅｒｎ．ｔｘｔとし、その内容を数９に示すものとする。

【０１６７】

【数９】

33

```
weather
news
http://www.asahi.com
http://www.sjmercury.com
http://www.cnn.com
http://naragw.sharp.co.jp/ZAURUS/
daily
```

34

＊【０１６８】ＵＮＩＸでは、このようなファイルオブジェクト名称の抽出作業は、ＯＳとともに提供されているコマンド群である数１０を用いて行なうことができる。

【０１６９】

【数１０】

＊

/usr/bin/fgrep　によるストリングパターンの抽出(パターンファイル指定可能)
/usr/bin/grep　によるストリングパターンの抽出
/usr/bin/grep －v によるストリングパターンの否定抽出
/usr/bin/awk　による特定のフィールド切りだし

【０１７０】たとえば、次のようなコマンドを入力することにより、ステップＳ２の処理を経たものがメモリ上の文字列バッファＵＲＬＢＵＦに出力される。

※

```
URLBUF='echo $LINE|grep -v "\?"|grep "\"GET"|grep 'H$'|
awk '{print $7}'|fgrep -f pattern.txt'
```

※【０１７１】

【数１１】

【０１７２】なお、上述したフィルタリングルールに反していれば、文字列バッファＵＲＬＢＵＦは空になる。この間の経過を以下に説明する。第１のｐｒｏｘｙプロセス１４により作成されたアクセスログファイル１５の☆

クライアント計算機名称－－[21/Aug/1995:09:27:43+0900] "GET http://naragw.sharp.co.jp/S2.gifHTTP/1.0" その他　キャッシュヒット状況

☆形式は既に述べたように次のとおりである。

【０１７３】

【数１２】

【０１７４】キャッシュヒット時には“H”という文字がキャッシュヒット状況として行の末尾に出力される。

【０１７５】まず、ｇｒｅｐ　－ｖ”＼?”に入力することで、?を含まない行を抽出し標準出力に出力する。それをさらにｇｒｅｐ”＼”ＧＥＴ”でその行がファイルオブジェクト取得要求であるＧＥＴコマンドである行のみ抽出し、標準出力に出力する。さらにｇｒｅｐ’Ｈ$’に入力することで行末が「H」で終了する、すなわちキャッシュにヒットしたものを選び標準出力に出力する。ここで、$は行の末尾を表現する正規表現である。

【０１７６】さらに空白がフィールド間の区切であると考えると、７番目のフィールドがファイルオブジェクト名称に相当するので、ａｗｋ’｛ｐｒｉｎｔ＄７｝’を使ってファイルオブジェクト名称のみを抽出している。

【０１７７】さらに、ファイルオブジェクト名称部分が、予め用意されたパターンファイルｐａｔｔｅｒｎ．ｔｘｔの中のいずれかの行に一致するか否かをｆｇｒｅｐ　－ｆ　ｐａｔｔｅｒｎ．ｔｘｔによりマッチングを行なってその結果を文字列バッファＵＲＬＢＵＦに出力する。

【０１７８】ステップＳ３では、文字列バッファＵＲＬＢＵＦが空でないか否かを判定する。もし文字列バッファＵＲＬＢＵＦが空の場合には、ステップＳ１に戻り、空でない場合にはステップＳ４へ進む。

【０１７９】ステップＳ４では、第２のｐｒｏｘｙプロセス３４によって、キャッシュ更新プロセス３６で抽出されたファイルオブジェクトをサーバ計算機１１から取得する。文字列バッファＵＲＬＢＵＦに格納されたキャッシュ更新アクセスの対象であるファイルオブジェクト名称に基づいて、第２のｐｒｏｘｙプロセス３４はサーバ計算機１１へアクセスするための子プロセスを起動する。子プロセスをキャッシュ更新子プロセスと呼ぶ。キャッシュ更新子プロセスは、第２のｐｒｏｘｙプロセス３４を通じてネットワーク接続をサーバ計算機１１に対して行なう。これにより、サーバ計算機１１に格納されるファイルオブジェクトへのアクセスが発生し、キャッシュファイル１６の内容が更新される。この処理の詳細は後述する。

【０１８０】ステップＳ５では、現在の時刻を調べ、指定時刻が過ぎていればプロセスを終了する。指定時刻を過ぎていなければステップＳ１に戻り、以上の処理を繰返す。

【０１８１】以上が、本願発明の第１の実施の形態におけるｐｒｏｘｙサーバ装置におけるアルゴリズムであ

る。図７は、ｐｒｏｘｙサーバ計算機５３のキャッシュ有効期限が２４時間で、サーバ計算機１１のファイルオブジェクトが同一名称でありながら、内容が３時間おきに更新される場合のファイルオブジェクトのデータがキャッシュされるようすを示している。

【０１８２】まず、ユーザアクセス（１）が発生したときに、キャッシュファイル１６に該当するファイルオブジェクトがなかったとすると、サーバ計算機１１からデータＢがアクセスされ、キャッシュファイル１６に書込まれるとともに、クライアント計算機２４に転送される。

【０１８３】次に、ユーザアクセス（２）が発生すると、キャッシュにヒットしたデータＢがクライアント計算機２４に転送される。このとき、サーバ計算機１１はデータＤになっているので、クライアント計算機２４は３時間程度古いステールデータを読んだことになる。しかし、この動作の結果アクセスログファイル１５にファイルオブジェクト名称とキャッシュにヒットしたという情報を含むアクセスログが記録されるので、図３を用いて説明した動作に従い、ｐｒｏｘｙサーバ装置５３が自発的にサーバ装置１１にアクセスし、データＥを読出しキャッシュファイル１６を更新する。

【０１８４】さらに、ユーザアクセス（３）が起きると、キャッシュされたデータＥがクライアント計算機２４に転送される。この動作をきっかけにしてアクセスログファイル１５にファイルオブジェクト名称とキャッシュにヒットしたという情報を含むアクセスログが記録されるので、図３を用いて説明した動作に従い、ｐｒｏｘｙサーバ装置５３が自発的にサーバ計算機１１にアクセス＊スし、データＩを読出してキャッシュファイル１６を更新する。

【０１８５】したがって、ユーザアクセス（４）が起きると、キャッシュされたデータＩがクライアント計算機２４に転送される。

【０１８６】以上の動作からわかるように、クライアント計算機２４はステールデータを読んでいることに変わりがないが、図２６の場合に比べると最新に近い情報を読んでいることになる。ユーザによるアクセスの間隔が３時間以内であれば、ステールデータは発生する確率が非常に低くなり、サーバ計算機１１の最新の情報を得ることが可能になることが理解できる。

【０１８７】なお、図３のステップＳ４のネットワークアクセス部の実現方法としては、キャッシュ更新プロセスから直接ＨＴＴＰプロトコルを発生させればよいが、ＵＮＩＸでは広く一般に利用されているｌｙｎｘ等のＷＷＷクライアントプログラムを利用すれば、ＨＴＴＰを発生するキャッシュ更新プロセスを新たに製作しなくてもＨＴＴＰに従ったネットワークアクセスを行なうキャッシュ更新プロセス操作を実現できる。その方法を以下に説明する。

【０１８８】ｌｙｎｘはＵＮＩＸ　ＯＳの上で動作するプログラムであるので、以下の説明ではＵＮＩＸの記述を使用する。

【０１８９】ｌｙｎｘでは次のコマンドを指定すれば、指定ＵＲＬを読出してｔｅｍｐ＿ｆｉｌｅというファイルに書込むことができる。

【０１９０】

【数１３】

```
lynx －source　http://www.xxx.co.jp/test/index.html　＞　temp_file
```

【０１９１】さらに、ｐｒｏｘｙプロセス経由のアクセスも、ＵＮＩＸ　ＯＳの環境変数ｈｔｔｐ＿ｐｒｏｘｙを設定することにより、ｌｙｎｘから利用可能である。また、ｐｒｏｘｙプロセスがＤｅｌｅＧａｔｅのようなｐｒｏｘｙソフトウェアであれば、ＵＲＬ表記にｐｒｏｘｙを指定することによりｐｒｏｘｙ経由のアクセスが可能である。これについては参考文献１２を参照されたい。

【０１９２】後者の方式を採用する場合について以下に説明する。ｐｒｏｘｙプロセスが、ネットワークアドレス※スがｐｒｏｘｙｓｅｒｖｅｒであるゲートウェイ計算機のＴＣＰ／ＩＰの１０００１番のポート番号を利用しているプロセスであるならば、このｐｒｏｘｙプロセス経由でサーバ計算機１１のファイルオブジェクトｈｔｔｐ：／／ｗｗｗ．ｘｘｘ．ｃｏ．ｊｐ／ｔｅｓｔ／ｉｎｄｅｘ．ｈｔｍｌにアクセスするためには次の数１４のコマンドを指定すればよい。

【０１９３】

【数１４】

```
lynx －source　http://proxyserver:10001/－_－http://www.xxx.co.jp/
test/index.html　＞　temp_file
```

【０１９４】したがって、本願発明を実施する際には、同様にしてｐｒｏｘｙプロセスを第２のｐｒｏｘｙプロセス３４のポート番号に設定すればよい。☆

☆【０１９５】

【数１５】

```
lynx －source　http://proxyserver:10001/－_－$URLBUF　＞　temp_file
```

【０１９６】本願発明では、このようなファイルオブジェクト（ＵＲＬ）のキャッシュ更新アクセスにより、第

1のｐｒｏｘｙプロセス１４の管理するキャッシュファイル１６を最新状態に保つことが目的である。したがって、ｔｅｍｐ＿ｆｉｌｅ自体は利用しないので捨ててよい。ＯＳがＵＮＩＸであれば、空ファイルとして／ｄｅｖ／ｎｕｌｌが準備されているので、このファイルに書込むようにすれば実際にはファイルの生成は行なわないので効率的である。

【０１９７】次に、第1のｐｒｏｘｙプロセス１４および第2のｐｒｏｘｙプロセス３４のキャッシュ管理手法について説明する。ｐｒｏｘｙプロセスのキャッシュ管理手法は、参考文献５、参考文献１２にあるような公知の技術であるキャッシュ制御機能を持ったＤｅｌｅＧａｔｅなどにおけるものである。ＤｅｌｅＧａｔｅにおけるキャッシュ管理手法は、ソースコードの形で公開されているので以下にその手法を述べる。なお、本願発明は＊

＊これと同様のキャッシュ管理手法が使われているキャッシュ機能付ｐｒｏｘｙサーバソフトウェアであれば、第1のｐｒｏｘｙプロセス１４および第2のｐｒｏｘｙプロセス３４として適用可能である。

【０１９８】上述したように、第2のｐｒｏｘｙプロセス３４がｐｒｏｘｙ．ｓｅｒｖｅｒという名称のシステムで動作しており、ＴＣＰ／ＩＰの１０００１番でＵＲＬ取得要求を受付けるものとする。このとき、文字列バッファＵＲＬＢＵＦに指定ＵＲＬ表記が格納されている場合に、キャッシュ更新プロセス３６は、数１６に記載のコマンドを実行することでｔｅｍｐ＿ｆｉｌｅに指定ＵＲＬのファイルオブジェクトが格納されると述べた。

【０１９９】

【数１６】

```
lynx －source　http://proxyserver:10001/－_－$URLBUF　＞　temp_file
```

【０２００】なお、第2のｐｒｏｘｙプロセス３４とキャッシュ更新プロセス３６の通信はＴＣＰ／ＩＰ、たとえばＵＮＩＸのメッセージ通信で行なわれる。

【０２０１】たとえば、文字列バッファＵＲＬＢＵＦにｈｔｔｐ：／／ｗｗｗ．ｓｈａｒｐ．ｃｏ．ｊｐ／ｓａｍｐｌｅ／ｔｅｓｔ．ｈｔｍｌという文字列が格納されているとする。キャッシュ更新プロセス３６が数１６のコマンドを実行すると、キャッシュ更新プロセス３６はＵＲＬ取得要求を第2のｐｒｏｘｙプロセス３４に出力する。第2のｐｒｏｘｙプロセス３４は、子プロセスを生成し、この子プロセスにＵＲＬ取得の処理を任せて自身のプロセスは再びキャッシュ更新プロセス３６からのＵＲＬ出力要求を待つ。キャッシュ更新プロセス３４による数１６の実行によりＵＲＬ取得を実行する第2のｐｒｏｘｙプロセス３４が生成した子プロセスは、図４のフローチャートに示す処理を行なう。

【０２０２】以下、第2のｐｒｏｘｙプロセス３４が生成した子プロセスの動作を図４のフローチャートを用いて説明する。

【０２０３】ｐｒｏｘｙプロセスでは、キャッシュファイル１６を持っているが、それは通常のＵＮＩＸファイルシステムの一部として作られる。キャッシュ用ディレクトリとして／ｃａｃｈｅというパーティションを使用するとすると、数１７に示すようにプロトコル名称がｈｔｔｐ、サーバ計算機名称ｗｗｗ．ｓｈａｒｐ．ｃｏ．ｊｐ、ファイルオブジェクトの実体部分がｓａｍｐｌｅ／ｔｅｓｔ．ｈｔｍｌとなる。

【０２０４】

【数１７】

```
http://www.sharp.co.jp/sample/test.html
```

【０２０５】このとき、キャッシュファイルは次の数１８に示す名称規則でキャッシュファイルを生成しようとする。

【０２０６】

【数１８】

```
/cache/ﾌﾟﾛﾄｺﾙ名称/ｻｰﾊﾞ計算機名称/ﾌｧｲﾙｵﾌﾞｼﾞｪｸﾄの実体部分
```

【０２０７】したがって、ｈｔｔｐ：／／ｗｗｗ．ｓｈａｒｐ．ｃｏ．ｊｐ／ｓａｍｐｌｅ／ｔｅｓｔ．ｈｔｍｌは数１９に示すファイルとなる（Ｓ１１）。

※【０２０８】

【数１９】

```
/cache/http/www.sharp.co.jp/sample/test.html
```

【０２０９】ｐｒｏｘｙプロセスはまず、数１９に示すファイル名称に変換し、そのファイルが存在するかどうかＯＳのｏｐｅｎ（ ）のシステムコールにより既存のファイルがオープンできるかどうかにより調べる（Ｓ１２）。ファイルが存在しなければ、ファイル新規作成モードで数１９に示すファイル名のキャッシュファイルを作成する（Ｓ１３）。また、キャッシュファイルに対して排他制御を行なって、他のプロセスが書込めないように排他ロックする（Ｓ１４）。

【０２１０】次に、ネットワークを経由してＵＲＬで指定されたファイルオブジェクトをサーバ計算機１１から取得する（Ｓ１５）。取得したファイルオブジェクトをクライアント計算機２４へ転送するとともにファイルオブジェクトを数１９に示すファイル名称のキャッシュファイルに書込む（Ｓ１６）。続いて、キャッシュファイルをクローズして排他ロックを解除する（Ｓ１７）。

【０２１１】上記処理では、キャッシュファイルを排他ロックしているが、この排他制御はＯＳがＵＮＩＸであればシステムコールであるｆｌｏｃｋ（　）関数を使って排他ロックできる。排他制御を使用する理由は、ｐｒｏｘｙプロセスは１つのファイルオブジェクトの転送ごとに子プロセスを起動してその子プロセスにキャッシュ制御を行なわせるため、同じキャッシュファイルシステムに対して子プロセス間で競合が発生する場合があるからである。

【０２１２】なお、キャッシュファイルに書込んだとき、ファイル変更時刻Ｔｍがファイル（数１９）の属性として記録される。ファイル更新時刻Ｔｍは、ＵＮＩＸＯＳのファイルシステムを管理するＯＳがファイルの属性として必ず付加するものである。すなわち、キャッシュファイルは通常のＯＳのファイルシステムをそのまま使用しており、何ら特殊なファイルシステムではない。

【０２１３】次に、キャッシュファイルが既に存在した場合を説明する（Ｓ１９～Ｓ２６）。

【０２１４】キャッシュファイル／ｃａｃｈｅ／ｈｔｔｐ／ｗｗｗ．ｓｈａｒｐ．ｃｏ．ｊｐ／ｓａｍｐｌｅ／ｔｅｓｔ．ｈｔｍｌの最終変更時刻ＴｍをＯＳから読出す。最終変更時刻は、ＵＮＩＸであれば、ｆｓｔａｔ（　）システムコールで得られるもので、過去から未来に向かって単調に増加する関数である（Ｓ１９）。

【０２１５】現在時刻をシステムコールから読出しＴｎｏｗという変数に格納する（Ｓ２０）。現在時刻Ｔｎｏｗは、ＯＳのシステムコールから得られる。最終変更時刻Ｔｍと、現在時刻Ｔｎｏｗを比較し、キャッシュ有効期限以内かどうかを判定する（Ｓ２１）。キャッシュ有効期限以内であれば、キャッシュファイルを共有排他ロックして他のプロセスからの書込はできないが、読出はできるようにする（Ｓ２２）。そして、既にあるキャッシュファイルを読出し、クライアント計算機２４に転送する（Ｓ２３）。さらに、キャッシュファイルをクローズし、共有排他ロックを解除する（Ｓ１７）。

【０２１６】また、キャッシュ有効期限を過ぎている場合は、キャッシュファイルに対して排他制御を行なって、他のプロセスが書込めないように排他ロックする（Ｓ２４）。次に、ネットワーク経由でＵＲＬで指定されたファイルオブジェクトをサーバ計算機１１から取得する（Ｓ２５）。取得したファイルオブジェクトをクライアント計算機２４へ転送するとともに、ファイルオブジェクトをキャッシュファイル（数１９）に書込む（Ｓ２６）。そして、キャッシュファイルをクローズして排他ロックを解除する（Ｓ１７）。

【０２１７】以上が、一般にソースコードが公開され、よく知られているｐｒｏｘｙプロセスのキャッシュ制御手順であるが、本願発明ではこの性質を利用して、キャッシュファイル１６を第１のｐｒｏｘｙプロセス１４と第２のｐｒｏｘｙプロセス３４で共有している。第２のｐｒｏｘｙプロセス３４は、キャッシュ有効期限０であるから、図３のフローチャートのＳ４により、第２のｐｒｏｘｙプロセス３４経由のサーバ計算機１１へのアクセスがキャッシュファイル１６の更新が必ず行なえることがわかる。

【０２１８】第１の実施形態においては、クライアント計算機２４の利用が始まる７時００分より起動し、クライアント計算機２４の利用が少なくなる夜２１時００分には終了するよう実施した。これにより、クライアント計算機２４の利用のない時間帯には、キャッシュ更新プロセス３６を排除し、当制御方式を採用した計算機のメモリ資源を無駄にしないようにしている。

【０２１９】以上説明したように、ｐｒｏｘｙサーバ計算機５３はユーザが行なったファイルオブジェクトに対するアクセス要求がキャッシュヒットすれば、その直後にサーバ計算機１１から当該ファイルオブジェクトを取得してキャッシュファイル１６を更新するので、別のユーザが同一ファイルオブジェクトをアクセスした場合に、最新のファイルオブジェクトが既にキャッシュファイル１６に格納されていることになる。したがって、キャッシュ有効期限を長めに取っておいても、キャッシュの有効期限よりも短い間隔で変更されるサーバ計算機のファイルオブジェクトがキャッシュに反映される確率が高くなる。

【０２２０】また、同一ユーザが時間をおいてから同一ファイルオブジェクトをアクセスし直すと、キャッシュの内容が最新のものに更新されていることになる。また、特定のパターンに一致するファイルオブジェクトのみ更新アクセスの対象となるので、予めファイルオブジェクトの更新頻度の高いサーバ計算機を登録しておけば不要なキャッシュ更新アクセスを削減でき、無駄なトラフィックを抑制して回線を有効に利用することができる。

【０２２１】［実施の形態２］次に、本願発明のゲートウェイ装置の第２の実施形態を説明する。この実施形態２は、実施形態１と同様であるが、実施形態１に対応するフローチャート（図３）のステップＳ４を改良したものである。実施形態１では、ステップＳ４においてアクセスログファイル１５からファイルオブジェクト名称を文字列格納バッファＵＲＬＢＵＦに格納して、第２のｐｒｏｘｙプロセス３４を介してサーバ計算機１１にアクセスしているが、第２の実施形態では処理の高速化を図るため複数のファイルオブジェクトに対するアクセスを、順不同で同時並列的に実行するものである。

【０２２２】本発明が実施できるＴＣＰ／ＩＰのネットワークでは、通信はパケット単位で行なわれるため、外部ネットワークの複数のサーバ計算機との同時通信が見掛け上可能である。したがって、ステップＳ４におけるキャッシュ更新子プロセスを複数同時に実施することが

できる。

【０２２３】内部ネットワーク２３には複数のクライアント計算機２４が接続されるので、ｐｒｏｘｙサーバ装置５３によるファイルオブジェクトの中継転送要求は、外部ネットワーク２５上の異なる複数のサーバ計算機１１へのアクセスが入り交じって行なわれる。したがって、ファイルオブジェクトに対するアクセスは外部ネットワーク２５上のさまざまな経路を通じて行なわれるため、実際のファイルオブジェクトの転送速度は、経路の途中のボトルネックによって決まる。

【０２２４】そのうちの１つのボトルネックは、ｐｒｏｘｙサーバ装置５３と外部ネットワーク２５を結ぶ最初の通信路（図１９における通信路１２７）である。さらには、外部ネットワーク２５上への複数のサーバ計算機１１への経路にはさまざまなボトルネックが存在する。このため、外部ネットワーク２５からｐｒｏｘｙサーバ計算機５３への転送速度は、この通信路の最大転送速度より小さい転送速度となるのが常である。たとえば、ｐｒｏｘｙサーバ装置５３と外部ネットワーク２５を結ぶ通信路（図１９における通信路１２７）の最大転送速度が６４ｋｂｐｓであっても、海外にあるサーバ計算機からの転送速度はその１０分の１以下であることがしばしばである。

【０２２５】したがって、ｐｒｏｘｙサーバ装置５３と外部ネットワーク２５を結ぶ最初の通信路１２７の最大転送容量に近い転送を行なうように、同時に複数のネットワーク接続を実施することで転送のスループットを上げることが可能になる。すなわちこのネットワーク接続を１個の子プロセスに対応させるとして、複数のキャッシュ更新子プロセスを起動し、この通信路１２７の転送容量が１００％近く使用されるように、複数のネットワーク接続を並列に実施した方が処理が高速化される。

【０２２６】そこで、同時に複数のキャッシュ更新アクセスを実行するように、キャッシュ更新子プロセスの最大並列実行数をＭＡＸＰＲＯＣＥＳＳ数と定める。これは、以下に説明するフローチャートではメモリ（図１９＊のメモリ２０６）に置かれる定数である。この値をあまり大きく取ると、ＯＳのプロセス手順などの管理資源（メモリ）を消費することになるので、システムの動作状況によって決める必要がある。この実施形態２では１０とした。

【０２２７】以下、具体的な処理手順を図５を参照しながら説明する。まず、変数としてメモリ２０６中にｐｒｏｃｅｓｓｅｓ変数を定義する。バックグラウンドで、現在走行しているキャッシュ更新子プロセスの数を求めてｐｒｏｃｅｓｓｅｓ変数に代入する。プロセスの数はＯＳのプロセス管理テーブルから得られる（Ｓ４２１）。次にｐｒｏｃｅｓｓｅｓ変数と指定された最大子プロセス数ＭＡＸＰＲＯＣＥＳＳを比較する。もし、ｐｒｏｃｅｓｓｅｓ≧ＭＡＸＰＲＯＣＥＳＳならばＳ４２３に移り、一定時間休止（ｓｌｅｅｐ）する（たとえば１０秒）。これは、先に走行しているキャッシュ更新子プロセスの処理を終了するのを待つために行なわれる（Ｓ４２３）。

【０２２８】次に、指定の終了時刻になったかどうかの判定を行なう。指定時刻を過ぎていれば、キャッシュ更新プロセスは処理を終了する。指定時刻を過ぎていなければ、Ｓ４２１以下の処理を繰返す（Ｓ４２４）。

【０２２９】また、Ｓ４２３でｐｒｏｃｅｓｓｅｓ＜ＭＡＸＰＲＯＣＥＳＳならば、第２のｐｒｏｘｙプロセス３４にアクセス要求を出力することによって、第２のｐｒｏｘｙプロセス３４はキャッシュ更新子プロセスをバックグラウンドプロセスとして１つ起動する（Ｓ４２５）。ＵＮＩＸ　ＯＳにおいて、プロセスをバックグラウンドで実行させるには、コマンドラインに＆の記号を付けて起動すればよい。文字列バッファＵＲＬＢＵＦには、Ｓ２により取得されたファイルオブジェクト名称が格納されているので、数２０に示すコマンドを実行すれば、キャッシュ更新子プロセスがバックグラウンドプロセスとして実施される。

【０２３０】

【数２０】

```
lynx －source　http://proxyserver:10001/－_－$URLBUF　＞　temp_file　＆
```

【０２３１】最後に、指定の終了時刻になったかどうかを判定し、指定時刻をすぎていればキャッシュ更新プロセスは処理を終了し、指定の終了時刻を過ぎていなければＳ１以下の処理を繰返す（Ｓ４２６）。

【０２３２】以上のような処理により、ＭＡＸＰＲＯＣＥＳＳ以下のプロセスで同時並列実行が可能になる。

【０２３３】この実施形態２により、キャッシュ更新処理速度を速め、常にキャッシュ更新処理を待つ状態にすることでキャッシュ更新処理の遅滞の拡大を防止することが可能となる。なお、パターンファイル３０を用意して、特定ファイルオブジェクトだけを更新対象にしているＳ２の処理も効果を上げている。

【０２３４】以上説明したように、実施の形態２では、キャッシュ更新処理を同時に行なうことで、キャッシュ更新の開始から終了までが短縮されるので、第１のｐｒｏｘｙプロセス１４によるユーザからのアクセス処理の頻度に追随してキャッシュ更新処理を進めることが可能となる。

【０２３５】〔実施の形態３〕実施の形態３においては、実施の形態２と同様に、キャッシュ更新子プロセスの並列実行を行なうが、さらにプロセスの制御を行なうことにより、内部ネットワーク２３のクライアント計算機２４を使用するユーザが第１のｐｒｏｘｙプロセス１４を使用して外部ネットワーク２５のサーバ計算機１１

に頻繁にアクセスしている時間帯に、キャッシュ更新プロセス３６を実行しても、キャッシュ更新子プロセスの数が抑制される。すなわち、キャッシュ更新子プロセスには低い優先権が与えられることで、クライアント計算機２４を使用するユーザのネットワーク利用を妨げないようにしたものである。

【０２３６】また、第１のｐｒｏｘｙプロセス１４を利用するユーザのネットワークアクセスは、通常１つのページが複数のファイルオブジェクトから構成されているため（たとえば、図２に示す情報ページ）、１つのページのアクセスにおいては、複数のファイルオブジェクトのアクセスが時間的に連続して発生する。したがって、キャッシュ更新アクセスの子プロセスの起動において、休止時間を設けて子プロセス起動を遅延させることにより、ユーザのページアクセスが終了してからキャッシュ更新アクセスが起きる確率を高めることができる。

【０２３７】また、同一のファイルオブジェクトに対するキャッシュ更新アクセスの条件が揃ったとき、すなわち、実施形態１および２のＵＲＬＢＵＦが空でないとき（図３および図４のＳ３がｙｅｓの場合）一定時間の無効時間を設け、前回キャッシュが更新されてから所定時間以内であればサーバ計算機１１に対するキャッシュ更新アクセスを行なわないことで、クライアント計算機２４が頻繁にファイルオブジェクトアクセスを行なっているときに、キャッシュ更新アクセスでサーバ計算機１１に対するアクセスが頻繁に発生することを防止するものである。

【０２３８】この処理の簡単な実現方法として、第２のｐｒｏｘｙプロセス３４のキャッシュ有効期限を０では＊なく、無効時間を設定して起動すればよい。この無効時間として、３０分または１時間等とし、サーバ計算機１１の変更頻度の平均の最小時間程度にすればよい。これにより、無効時間以内に実施形態１、２または３の方式でキャッシュ更新アクセスを行なっても、キャッシュにヒットすることになるのでサーバ計算機１１にアクセスが発生しなくなる。

【０２３９】図６は、実施の形態３の処理を示すフローチャートである。図６は、実施形態２の処理を示すフローチャートである図５のＳ４２１とＳ４２５をＳ４３１とＳ４３５に置き換えたものである。

【０２４０】図６に示す処理を行なうに先立って、最大ネットワーク接続個数ＭＡＸＰＲＯＣＥＳＳという定数を定める。この定数は、キャッシュ更新子プロセスの数と第１のｐｒｏｘｙプロセス１４がネットワークアクセスの中継のために起動した子プロセスの数を加算した値の最大を示す定数である。したがって、ｐｒｏｘｙサーバ装置５３は、なるべくのＭＡＸＰＲＯＣＥＳＳ以下にサーバ計算機１１へのネットワークアクセスを制限しようとする。図６のＳ１～Ｓ３の処理は、実施形態１、２と同様であるのでここでの説明は繰返さない。

【０２４１】バックグラウンドで走行しているキャッシュ更新子プロセスの数と、第１のｐｒｏｘｙプロセス１４のネットワークアクセス中継子プロセスの数を求めて加算してｐｒｏｃｅｓｓｅｓ変数（数２１）とする（Ｓ４３１）。

【０２４２】

【数２１】

processes　<==　ｷｬｯｼｭ更新子ﾌﾟﾛｾｽの数＋第1proxyのﾈｯﾄﾜｰｸｱｸｾｽ中継子ﾌﾟﾛｾｽの数

【０２４３】ｐｒｏｃｅｓｓｅｓ変数と指定された最大子プロセスＭＡＸＰＲＯＣＥＳＳを比較し、ｐｒｏｃｅｓｓｅｓ≧ＭＡＸＰＲＯＣＥＳＳならばＳ４３３へ移り、一定時間休止（ｓｌｅｅｐ）する（たとえば１０秒）。これは、先に起動されたキャッシュ更新子プロセス数の処理が終了するのを待つものである。

【０２４４】次に、指定の終了時刻になったか否かを判定し、指定時刻を過ぎていればキャッシュ更新プロセスは処理を終了する。指定の終了時刻になっていなければ、Ｓ４３１に戻って処理を続行する（Ｓ４３４）。また、Ｓ４３２でｐｒｏｃｅｓｓｅｓ＜ＭＡＸＰＲＯＣＥＳＳならば、抽出したファイルオブジェクト名称（ＵＲＬＢＵＦに格納されたもの）をもとに、第２のｐｒｏｘｙプロセス３６を用いてアクセスするキャッシュ更新子※※プロセスをバックグラウンドプロセスとして１つ起動する。

【０２４５】このとき、子プロセスは数２２に示すように、休止時間をＤＥＬＡＹＴＩＭＥ変数（単位は秒）としてＯＳの休止機能をｓｌｅｅｐを使ってＤＥＬＡＹＴＩＭＥ秒間休止してからキャッシュ更新アクセスを行なうようにする。１ページＷＷＷデータは、図２に示すように複数の画像ファイルオブジェクトからなる場合、キャッシュヒット時は１０秒以下でｐｒｏｘｙサーバ装置５３からクライアント計算機２４に転送可能であることが経験的に知られているので、ＤＥＬＡＹＴＩＭＥは１０秒程度としている。

【０２４６】

【数２２】

(sleep　$DELAYTIME ; lynx －source http:// proxyserver:10001/－_－$URLBUF

＞temp_file)&

【０２４７】最後に、指定の終了時刻になったかどうかを判定し、指定時刻を過ぎていればキャッシュ更新プロ

セスは処理を終了し、指定終了時刻を過ぎていなければ、Ｓ１へ戻り処理を繰返す（Ｓ４３６）。

【０２４８】以上のように、キャッシュ更新子プロセスの個数と第１のｐｒｏｘｙプロセス１４を利用しているユーザのネットワークアクセスの中継子プロセス数の合計を制限しながらキャッシュ更新子プロセスを制御する。この結果、第１のｐｒｏｘｙプロセス１４を利用しているユーザによるネットワークアクセスの数が増えると、キャッシュ更新子プロセスは起動されにくくなる。したがって、ユーザが第１のｐｒｏｘｙプロセス１４を利用している時間帯に、キャッシュ更新プロセス３６を実行しても、キャッシュ更新子プロセスには低い優先権が与えられることで、ユーザのネットワーク活動を妨げないようにすることができる。

【０２４９】また、ユーザのファイルオブジェクトに対するアクセスがキャッシュにヒットした場合は、通常１ページの転送は、数秒以内に終了するが、キャッシュ更新子プロセスはＤＥＬＡＹＴＩＭＥ（秒）遅延してから起動されるので、ＤＥＬＡＹＴＩＭＥを１０秒程度に設定すればユーザのファイルオブジェクトアクセスとキャッシュ更新アクセスは時間的にずれて実施される（すなわちパイプライン）ので、ｐｒｏｘｙサーバ計算機５３への負担を減らし、ユーザに快適なネットワーク利用を提供することが可能となる。

【０２５０】以上説明したように、実施の形態３ではユーザが第１のｐｒｏｘｙプロセス１４を利用している時間帯に、キャッシュ更新プロセス３６を実行しても、キャッシュ更新子プロセスには低い優先権が与えられることで、ユーザのネットワーク利用を妨げない効果がある。また、遅延時間を導入することで、キャッシュ更新処理はユーザの連続したファイルオブジェクト転送処理が終了してから行なわれるので、ｐｒｏｘｙサーバ計算機５３の処理負担を軽減し、ユーザへのファイルオブジェクト転送を最優先できる。また、無効時間が設定されるため、クライアント計算機２４が無効時間以内に同一ファイルオブジェクトに対してアクセスを行ない、キャッシュ更新アクセスの条件が揃ってもキャッシュ更新アクセスはキャッシュにヒットすることになり、実質的にサーバ計算機１１に対するアクセスが発生しない。これにより、クライアント計算機２４が同一ファイルオブジェクトを頻繁にアクセスしても、時間のかかるサーバ計算機１１へのキャッシュ更新アクセスが抑制されるのでサーバ計算機１１への負担の軽減とキャッシュ更新処理の効率化が図れる。

【０２５１】［実施の形態４］実施の形態１～３は、ｐｒｏｘｙサーバ計算機に関するものであったが、実施の形態４ではファイルオブジェクトの中継制御のみを行なうゲートウェイ装置（ｒｅｄｉｒｅｃｔｏｒ計算機）について説明する。図８は、キャッシュサーバ計算機（ｐｒｏｘｙサーバ計算機）６５～６７およびｒｅｄｉｒｅｃｔｏｒ計算機７２～７７をローカルネットワーク７１で接続する構成を示す図である。

【０２５２】ｒｅｄｉｒｅｃｔｏｒ計算機７２～７７は、ｐｒｏｘｙサーバ計算機の１種でもあるが、実施の形態４のｒｅｄｉｒｅｃｔｏｒ計算機７２～７７は、ｐｒｏｘｙサーバ計算機６５～６７への中継のみを行ないキャッシュは行なわない。

【０２５３】以下、ｒｅｄｉｒｅｃｔｏｒ計算機７２～７７の動作について詳細に説明する。

【０２５４】ｒｅｄｉｒｅｃｔｏｒ計算機７２～７７は、複数のｐｒｏｘｙサーバ計算機６５～６７とクライアント計算機群との間に介在し、ｐｒｏｘｙサーバ計算機６５～６７とはローカルネットワーク７１で接続され、クライアント計算機群とはローカルネットワーク７８で接続される。また、ファイルオブジェクトを提供するサーバ計算機１１とｐｒｏｘｙサーバ計算機６５～６７とは、広域ネットワーク６１で接続される。

【０２５５】まず、クライアント計算機２４からのファイルオブジェクトに対する読出要求をローカルネットワーク７８を介してｒｅｄｉｒｅｃｔｏｒ計算機７３内のｒｅｄｉｒｅｃｔｏｒプロセスが受信する。ｒｅｄｉｒｅｃｔｏｒ計算機は、上述した図１７～１９に示すワークステーション２００～４００で実現可能である。すなわち、ｒｅｄｉｒｅｃｔｏｒプロセスに相当するプログラムをメモリ２０６内に格納し、そのプログラムをＣＰＵ２０２が実行することによってこのｒｅｄｉｒｅｃｔｏｒ計算機の機能が実現される。ｒｅｄｉｒｅｃｔｏｒプロセスは、受信した読出要求の内容に基づいてハッシュ関数の計算を行ないその計算結果に従って中継するｐｒｏｘｙサーバ計算機６５～６７を選択する。そしてｒｅｄｉｒｅｃｔｏｒプロセスは、読出要求を選択されたｐｒｏｘｙサーバ計算機６５～６７内のｐｒｏｘｙプロセスに中継する。

【０２５６】本発明を非常に有効に適用できる例として、前述したインターネット上のWWWシステムがある。図９は、WWWシステム上でのｒｅｄｉｒｅｃｔｏｒ計算機の処理手順を示すフローチャートである。

【０２５７】WWWシステムではネットワーク上に分散したファイルオブジェクト名称はUniform Resource Locator（ＵＲＬ）と呼ばれる形式で表現され特定される。

【０２５８】たとえば、ｒｅｄｉｒｅｃｔｏｒ計算機７３が、クライアント計算機２４からの読出要求を受信したとすると、ｒｅｄｉｒｅｃｔｏｒ計算機７３は要求に含まれるＵＲＬを取出す（Ｓ３１）。取出されたＵＲＬに基づいてハッシュ関数による計算を行なう。ここで、ハッシュ関数にはさまざまなものが考えられるが、実施の形態４ではＵＲＬ形式で表現されたファイルオブジェクト名称に含まれる文字の文字コードをすべて加算し、ｐｒｏｘｙサーバ計算機６５から６７の台数で割った剰余を使用した（Ｓ３３）。

【０２５９】このハッシュ関数によって算出された値によって、読出要求を中継するｐｒｏｘｙサーバ計算機６５～６７の番号を決定する（Ｓ３４）。そして、選択されたｐｒｏｘｙサーバ計算機６５～６７に読出要求が中継される（Ｓ３５）。

【０２６０】同一のファイルオブジェクトの読出要求は、すべて同一のｐｒｏｘｙサーバ計算機６５～６７で中継されることが、キャッシュの有効利用の観点からは望ましい。なぜなら、同一のファイルオブジェクトを異なるｐｒｏｘｙサーバ計算機で重複して保持すると、キャッシュ容量が減少し、キャッシュファイルの有効利用が図れなくなるからである。そのためには、ハッシュ関数がすべてのｒｅｄｉｒｅｃｔｏｒ計算機７２～７７内のｒｅｄｉｒｅｃｔｏｒプロセスで同一のものである必要がある。

【０２６１】上述したハッシュ関数は、各ｐｒｏｘｙサーバ計算機６５～６７が均等に選択されるため、それぞれのｐｒｏｘｙサーバ計算機６５～６７に対する負荷は均等になるが、適切にハッシュ関数を選ぶことにより、各ｐｒｏｘｙサーバ計算機６５～６７の能力に応じて意図的に均等でなく負荷を分散することも容易に実現できる。すなわち、処理能力の高いｐｒｏｘｙサーバ計算機に対しては、選択される可能性を高くし、処理能力の低＊いｐｒｏｘｙサーバ計算機に対しては、選択される可能性を低くすることによって各ｐｒｏｘｙサーバ計算機の処理速度を均等にするものである。

【０２６２】このようにして算出された番号により示されるｐｒｏｘｙサーバ計算機が、選択されたｐｒｏｘｙサーバ計算機となり、ｒｅｄｉｒｅｃｔｏｒプロセスは選択されたｐｒｏｘｙサーバ計算機内のｐｒｏｘｙプロセスにクライアント計算機２４からの読出要求を中継する。ｐｒｏｘｙサーバ計算機６５～６７内のｐｒｏｘｙプロセスは、ｒｅｄｉｒｅｃｔｏｒ計算機７３内のｒｅｄｉｒｅｃｔｏｒプロセスにより中継されたクライアント計算機２４からの読出要求を広域ネットワーク６１を介してサーバ計算機１１に中継し、転送されたファイルオブジェクトをキャッシュする。

【０２６３】ｒｅｄｉｒｅｃｔｏｒプロセスは、広く一般に利用されているＤｅｌｅＧａｔｅを利用して実施することが可能である。実施の形態４のｒｅｄｉｒｅｃｔｏｒプロセスは、ＤｅｌｅＧａｔｅの中継記法とフィルタ機能を利用して実現されている。以下に、中継記法の例を示す。

【０２６４】

【数２３】

http://proxyserver:10080/－_－http://www.xxx.co.jp/test/index.html

【０２６５】数２３の記述のうち、最初のｈｔｔｐは使用するプロトコル名を示し、ｐｒｏｘｙｓｅｒｖｅｒはネットワーク上のｐｒｏｘｙサーバ計算機のアドレスを示すものである。また、１００８０はＴＣＰ／ＩＰ接続のポート番号と呼ばれ、ｐｒｏｘｙサーバ計算機上のどのプロセスに対し読出要求を送るのかを示す。－_－は、これがＤｅｌｅＧａｔｅの中継記法であることを示す。残りのｈｔｔｐ：／／ｗｗｗ．ｘｘｘ．ｃｏ．ｊｐ／ｔｅｓｔ／ｉｎｄｅｘ．ｈｔｍｌが、本来の読出要求のＵＲＬである。

【０２６６】ＤｅｌｅＧａｔｅが上記読出要求を受信すると、ｐｒｏｘｙサーバ計算機ｐｒｏｘｙｓｅｒｖｅｒのポート番号１００８０で示されるｐｒｏｘｙプロセスに対し、ＵＲＬで示されるファイルオブジェクトの読出要求を中継する。すなわち、読出要求の中で、中継するｐｒｏｘｙサーバ計算機を指定することが可能である。

【０２６７】フィルタ機能は、外部プログラムを利用してＤｅｌｅＧａｔｅ機能を拡張するための仕組みである。ＤｅｌｅＧａｔｅはクライアント計算機もしくは他のｐｒｏｘｙサーバ計算機からの読出要求を受信すると、受信した読出要求を一旦外部プログラムに引渡し、この外部プログラムによって処理された読出要求を再び受取って中継等の処理を行なうことが可能である。

【０２６８】図１１は、ｒｅｄｉｒｅｃｔｏｒ計算機の内部構成を示す図である。実施の形態４において、ＤｅｌｅＧａｔｅによって実現されるｒｅｄｉｒｅｃｔｏｒプロセス８１が、たとえば、数２４に示すＵＲＬを含む読出要求を受信したとする。

【０２６９】

【数２４】

http://www.xxx.co.jp/test/index.html

【０２７０】これは、ｗｗｗ．ｘｘｘ．ｃｏ．ｊｐで表わされるサーバ計算機１１にこの読出要求を送信することを示している。

【０２７１】しかし、実施の形態４ではＤｅｌｅＧａｔｅのフィルタ機能により、外部プログラムとして実現されたｐｒｏｘｙ切換制御部８２においてＵＲＬをもとにハッシュ関数の計算を行ない（図９のＳ３２～Ｓ３３）、その演算結果によってｐｒｏｘｙサーバ計算機ｐｒｏｘｙｓｅｒｖｅｒ１が選択されたとすると、上記読出要求内のＵＲＬを数２５のように書替える（図９のＳ３４）。

【０２７２】

【数２５】

http://proxyserver1:10080/－_－http://www.xxx.co.jp/test/index.html

【０２７３】外部プログラムであるｐｒｏｘｙ切換制御部８２によって書替えられた読出要求を受取ったｒｅｄｉｒｅｃｔｏｒプロセス８１は、ｐｒｏｘｙｓｅｖｅｒ１で表わされるｐｒｏｘｙサーバ計算機にｗｗｗ．ｘｘｘ．ｃｏ．ｊｐで表わされるサーバ計算機１１に対する読出要求を中継する（図９のＳ３５）。

【０２７４】したがって、ハッシュ関数で計算された結果に基づいてＵＲＬ中のｐｒｏｘｙサーバ計算機のアドレスをｐｒｏｘｙｓｅｒｖｅｒ１に変えることにより、元の読出要求のＵＲＬに基づいて中継するｐｒｏｘｙサーバ計算機を選択することができる。実際に、２台のｐｒｏｘｙサーバ計算機を用意し、ハッシュ関数はＵＲＬに含まれるファイルオブジェクト名称の文字の文字コードをすべて加算し（図８のＳ３２）、ｐｒｏｘｙサーバ計算機の台数２で割った剰余を使用した（図８のＳ３３）。すなわち、ハッシュ関数の計算結果が０であった場合ｐｒｏｘｙｓｅｒｖｅｒ０により中継し、１であった場合はｐｒｏｘｙｓｅｒｖｅｒ１で中継を行なうものとする。

【０２７５】図１０は、図２に示す同一ページのファイルオブジェクトが、２台のｐｒｏｘｙサーバ計算機に分散されるようすを示したものである。図１０の上段は、本来の読出要求のＵＲＬ、ファイルオブジェクト名称の文字コードの総計（ｃｈｅｃｋ　ｓｕｍ）、文字コードの総計をｐｒｏｘｙサーバ計算機の台数である２で割った余りおよび選択されたｐｒｏｘｙサーバ計算機のアドレスを示している。図１０の下段は、ｐｒｏｘｙ切換制御部８２で書替られたＵＲＬを示している。この例では、ハッシュ関数の結果が０の方が多くなっており、ｐｒｏｘｙｓｅｒｖｅｒ０への中継が多くなってしまうが、これはサンプル数が少ないことによるものである。実際の運用では膨大なアクセスが、ハッシュ関数による分散が確率的に均等になるという性質のもとに、ほぼ２分の１ずつ均等に分散されることが確認されている。

【０２７６】以上説明したように、複数あるｐｒｏｘｙサーバ計算機が別々な経路で広域ネットワークに接続されている場合、クライアント計算機からのアクセス要求を複数あるｐｒｏｘｙサーバ計算機に分散することは、複数ある経路に分散することでもある。したがって経路の有効利用が可能となり、スループットが向上する。ｐｒｏｘｙサーバ計算機の処理能力には限界があるが、クライアント計算機からのアクセス要求が複数のｐｒｏｘｙサーバ計算機に分散されることにより、ｐｒｏｘｙサーバ計算機１台で処理されるアクセス要求数が減少し、スループットが向上する。

【０２７７】また、同一ページ内に複数のファイルオブジェクトが含まれている場合、それらのファイルオブジェクトに対するアクセス要求も複数のｐｒｏｘｙサーバ計算機によって分散して処理されるので、単一ページのアクセス速度も向上する。

【０２７８】また、どのクライアント計算機からの中継要求であっても、同一ＵＲＬに対するハッシュ関数が同じであるので、常に同じｐｒｏｘｙサーバ計算機が選択されることになり、同一ＵＲＬのアクセスが２回以上あればキャッシュによるアクセスの高速化が図れる。さらに、ハッシュ関数が一意にきまるものであれば任意の関数が選択できるので、ｐｒｏｘｙサーバ計算機の数もまた任意の台数が使用できる。

【０２７９】実施の形態４において、ハッシュ関数内で使われているＵＲＬに含まれる文字の文字コードをすべて加算した値（ｃｈｅｃｋ　ｓｕｍ）は、平均して数千程度になることが確かめられている。たとえば、シャープ奈良工場における実施においては、１９９６年３月１３日に実測された３０４８１件のファイルオブジェクトに対するアクセス要求のＵＲＬに含まれる文字や文字コードの合計の平均は４４３８（小数点以下四捨五入）であった。

【０２８０】前述したとおり、実施の形態４ではこのアクセス要求のＵＲＬに含まれる文字や文字コードの合計をｐｒｏｘｙサーバ計算機の台数で割った剰余をハッシュ関数として使用し、ｐｒｏｘｙサーバ計算機の選択に使用している。したがって、図１０の例ではｐｒｏｘｙサーバ計算機は２台であるが、この台数が数千程度であっても、ハッシュ関数によるｐｒｏｘｙサーバ計算機の選択はほぼ均等に行なわれると考えられる。

【０２８１】また、同一のハッシュ関数であれば、ｒｅｄｉｒｅｃｔｏｒ計算機の数は任意である。したがって、本方式ではスケーラビリティも非常に優れている。

【０２８２】また、本方式ではキャッシュ付のｐｒｏｘｙサーバ計算機は既存のものをそのまま使用することが可能である。したがって、図２４の従来の方式に比べ導入が容易で経済的である。また、クライアント計算機に関しても、既存のクライアント計算機を使用して、ｒｅｄｉｒｅｃｔｏｒ計算機を既存のｐｒｏｘｙサーバ計算機に見立てて設定を行なうだけでよく、特殊なクライアント計算機を用意する必要はない。

【０２８３】［実施の形態５］実施の形態５は、実施の形態４において独立した計算機上で実行していたｒｅｄｉｒｅｃｔｏｒプロセスを、クライアント計算機２４上で実行する。図１２は、実施の形態５における装置の内部構成を示す図であり、クライアント計算機２４内のクライアントプロセス８４（図８のクライアント計算機２４と同じ動作をするプロセス）からの読出要求は、ローカルネットワーク上には送出されず、クライアント計算機２４内部のｒｅｄｉｒｅｃｔｏｒプロセス８５に直接送信される。この送信は、たとえばＵＮＩＸのメッセージ通信によって行なわれる。

【０２８４】ｒｅｄｉｒｅｃｔｏｒプロセス８５がアクセス要求を受信した後の処理は、上述したｒｅｄｉｒｅｃｔｏｒ計算機７２～７７内で動作するｒｅｄｉｒｅｃ

ｔｏｒプロセス８１と同様であり、ローカルネットワークを介する他のクライアント計算機からのアクセス要求を受信して処理することも可能である。したがって、ここでの詳細な説明は繰返さない。また、このｒｅｄｉｒｅｃｔｏｒプロセス８５の機能をクライアントプロセス８４のプログラムに直接組込むことも可能である。

【０２８５】図８に示す実施の形態４のクライアント計算機２４とｒｅｄｉｒｅｃｔｏｒ計算機７２～７７との通信が、実施の形態５では同じクライアント計算機２４内部で処理されるため、ローカルネットワーク上での通信量が削減できる。したがって、独立したｒｅｄｉｒｅｃｔｏｒ計算機を用意する必要がなく、クライアント計算機２４を利用してｒｅｄｉｒｅｃｔｏｒ計算機の機能を実現できるためコストの削減が可能となる。

【０２８６】［実施の形態６］実施の形態６では、実施の形態４において独立した計算機上で実行していたｒｅｄｉｒｅｃｔｏｒプロセスを、ｐｒｏｘｙサーバ計算機上で実行する。図１３は、実施の形態６における装置の内部構成を示す図であり、たとえば、クライアント計算機２４ｂから読出要求があった場合に、その読出要求はｐｒｏｘｙサーバ計算機９９内のｒｅｄｉｒｅｃｔｏｒプロセス９１で受信される。その後、ｒｅｄｉｒｅｃｔｏｒプロセス９１は、図９を用いて説明したようにアクセス要求を中継するｐｒｏｘｙサーバ計算機を選択し、その選択されたｐｒｏｘｙサーバ計算機が９８であれば、ｐｒｏｘｙサーバ計算機９８のｐｒｏｘｙプロセス８６に対してアクセス要求を中継する（９２）。また、受信したｒｅｄｉｒｅｃｔｏｒプロセスの動作しているｐｒｏｘｙサーバ計算機と選択されたｐｒｏｘｙサーバ計算機がともに９９である場合、読出要求の中継はネットワーク上に送出されずｐｒｏｘｙサーバ計算機９９内部で直接行なわれる（９３）。そして、ｐｒｏｘｙプロセス（８６または８９）が広域ネットワーク６１を介してサーバ計算機１１に対してアクセス要求を送出する（６２または６３）。

【０２８７】同様に、クライアント計算機２４ａから読出要求があった場合に、その読出要求は、ｐｒｏｘｙサーバ計算機９８上のｒｅｄｉｒｅｃｔｏｒプロセス９６で受信される。その後、ｒｅｄｉｒｅｃｔｏｒプロセスは、ｐｒｏｘｙサーバ計算機の選択を行ない、その選択されたｐｒｏｘｙサーバ計算機が９９であれば、ｐｒｏｘｙサーバ計算機９９のｐｒｏｘｙプロセス８９に対してアクセス要求を中継する（９５）。

【０２８８】また、受信したｒｅｄｉｒｅｃｔｏｒプロセスの動作しているｐｒｏｘｙサーバ計算機と選択されたｐｒｏｘｙサーバ計算機がともに９８である場合、読出要求の中継はネットワーク上に送出されずｐｒｏｘｙサーバ計算機９８内部で直接行なわれる（９６）。そして、ｐｒｏｘｙプロセス（８６または８９）が広域ネットワーク６１を介してサーバ計算機１１に対してアクセス要求を送出する（６２または６３）。

【０２８９】サーバ計算機１１から取得したファイルオブジェクトをキャッシュファイル（８７または９０）に格納する処理は、上述した処理と同様であるので詳細な説明は繰返さない。

【０２９０】また、このｒｅｄｉｒｅｃｔｏｒプロセスの機能を、ｐｒｏｘｙサーバ計算機上でｐｒｏｘｙサーバとしての機能を実現しているプログラムに直接組込むことも可能である。

【０２９１】したがって、ｒｅｄｉｒｅｃｔｏｒ計算機とｐｒｏｘｙサーバ計算機の通信の一部がｐｒｏｘｙサーバ計算機内部で処理されるため、ローカルネットワーク上での通信量が削減できる。また、独立したｒｅｄｉｒｅｃｔｏｒ計算機を用意する必要がなく、ｐｒｏｘｙサーバ計算機を利用してｒｅｄｉｒｅｃｔｏｒ計算機の機能を実現できるためコストを削減することが可能になる。

【０２９２】［実施の形態７］実施の形態４のｒｅｄｉｒｅｃｔｏｒ計算機７２～７７はアクセス要求の中継のみを行なっていた。しかし、図２３で説明したようなツリー状の構成が有効であるような大規模なローカルネットワークでは、１次キャッシュｐｒｏｘｙサーバ計算機５０３～５０８の持つキャッシュ機能をｒｅｄｉｒｅｃｔｏｒ計算機に持たせることによって、ネットワーク上の通信量およびｐｒｏｘｙサーバ計算機の負荷を軽減することが可能となる。図１４は、そのシステム構成を示す図である。ｒｅｄｉｒｅｃｔｏｒサーバ計算機４０～４５は、図１１に示すｒｅｄｉｒｅｃｔｏｒ計算機８０と図１６に示すｐｒｏｘｙサーバ計算機１３を組合せることにより可能である。すなわち、図１６のリード要求１７を図１１のｒｅｄｉｒｅｃｔｏｒプロセス８１が受信することによって実現可能であるので、ここでの詳細な説明は繰返さない。また、実施の形態５におけるクライアント計算機内部でキャッシュを行なうことによっても、ローカルネットワーク上の通信量の削減とｐｒｏｘｙサーバ計算機の負荷を軽減することが可能となる。

【０２９３】上述したように、ｒｅｄｉｒｅｃｔｏｒ計算機においてキャッシュすることにより、ローカルネットワーク上のでの通信量およびｐｒｏｘｙサーバ計算機の負荷を軽減することができる。

【０２９４】［実施の形態８］上述した実施の形態４から７は、排他的に使用されなければいけないものではない。すなわち、実施の形態８においてはｒｅｄｉｒｅｃｔｏｒ機能を実現するすべての計算機（ｐｒｏｘｙサーバ計算機、クライアント計算機、ｒｅｄｉｒｅｃｔｏｒ計算機）において、ｒｅｄｉｒｅｃｔｏｒ機能の使用するハッシュ関数が同一であり、すべてのアクセス要求がそれぞれ同一のｐｒｏｘｙサーバ計算機等のキャッシュ機能を有する計算機で中継およびキャッシュされれば、実施の形態４～７に記載された各種の計算機は、単一の

53

ネットワーク内に混在して使用されても全く問題はない。したがって、それぞれの会社等の組織におけるさまざまなネットワークや計算機の運用事情に応じ、非常に柔軟なネットワーク運用を行なうことが可能となる。

【０２９５】ある程度以上の大規模なローカルネットワークでは、広域ネットワークとの接続が複数の経路（図８の６２～６４）からなる場合もある。そのような場合、ｐｒｏｘｙサーバ計算機ごとに異なる経路を使用すると、クライアント計算機からのアクセス要求を複数ある経路に分散して中継することになる。したがって、複数ある広域ネットワークとの通信経路を有効利用し、広域ネットワークとローカルネットワークとの通信のスループット向上が図れる。

【０２９６】実際の運用においては、すべての計算機の設定を一度に変えることは現実的ではない。移行には暫く時間がかかることが普通である。より効率のよい運用を図るためには、すべてのｒｅｄｉｒｅｃｔｏｒ機能を有する計算機内でハッシュ関数を一致させる必要があるため、すべてのクライアント計算機がｒｅｄｉｒｅｃｔｏｒ計算機を利用するようシステムを構築するか、クライアント計算機がｒｅｄｉｒｅｃｔｏｒ機能を持つ必要がある。しかし、そのようなシステムを構築しなくても＊

54

＊ｒｅｄｉｒｅｃｔｏｒ機能を有する計算機を使用することは可能である。すなわち、本発明の各方式では、従来のｐｒｏｘｙサーバ計算機やクライアント計算機を全く手を付けない状態で、システムの一部にｒｅｄｉｒｅｃｔｏｒ機能を持たせて、システム全体はそのまま使用することも可能である。

【０２９７】同様に、システムの再構築の際、複数のｒｅｄｉｒｅｃｔｏｒ機能におけるハッシュ関数が一致しない場合もあり得る。このような場合でも、システム全体の動作自体には影響はないので、通常と同様に利用が可能である。上述したいずれの場合も利用は可能であるが、ｐｒｏｘｙサーバ計算機その他の有効利用の観点からは望ましい状況とはいえない。したがって、できるだけ早い時期にすべてのｒｅｄｉｒｅｃｔｏｒ機能を有する計算機での設定を統一させることが望ましい。しかし移行期に新旧の設定を混在していても不必要な混乱を起こさず、スムーズに移行することが可能である。

【０２９８】なお、ｈｔｔｐプロトコルを含めた参考文献リストを以下に示す。

【０２９９】

【数２６】

・[Ref. 1]横塚実、『インターネットとはなにか』
http://www.rwcp.or.jp/people/yokotuka/internet_report.html

・[Ref. 2]page170, "ｷｬｯｼｭ利用でﾄﾗﾌｨｯｸを低減するﾌﾟﾛｷｼｻｰﾊﾞｰ"
『企業ﾕｰｻﾞのためのｲﾝﾀｰﾈｯﾄﾊﾝﾄﾞﾌﾞｯｸ』日経ｺﾐｭﾆｹｰｼｮﾝ、
ISBN4－822－1365－X, 日経BP社

・[Ref. 3]WWWに関する参考文献
http://www.w3.org/hypertext/WWW/Protocols/Overview.html

・[Ref. 4]HTTPの仕様に関するインターネット上の文献
http://www.w3.org/hypertext/WWW/Protocols/HTTP/HTTP2.html

・[Ref. 4の日本語資料]日本語のHTTPプロトコル仕様書としては、木内貴弘
(kiuchi@epistat.m.u－tokyo.ac.jp)氏翻訳の
http://www.epistat.m.u－tokyo.ac.jp/internet/draft－fielding－http－spec－01
－jp.txt、ｵﾘｼﾞﾅﾙﾌｧｲﾙ名：著者：T.Berners-Lee、R.T.Fielding、H.Frystyk
Nielsenがある。

・[Ref. 5]電子技術総合研究所の佐藤氏の作成されたdelegateについては
http://www.etl.go.jp:8080/etl/People/ysato@etl.go.jp/DeleGate/から情報が
入手できる。

・[Ref. 6]松下電器産業株式会社、特開平6－290090、"遠隔ﾌｧｲﾙｱｸｾｽ
ｼｽﾃﾑ

・[Ref. 7]WWWｼｽﾃﾑ全般の情報は、World Wide Webｺﾝｿｰｼｱﾑ
http://www.w3.org/hypertext/WWW/TheProject/

【０３００】

【数２７】

・[Ref. 8]CERN httpdに関しては
http://www.w3.org/hypertext/WWW/Daemon/
・[Ref. 9]大家隆弘(徳島大学), WWW 多段proxy計画(案)
http://www.orions.ad.jp/project/multiproxy/multiproxy-jp.html
・[Ref. 10]CERNhttpdでのキャッシュ制御解説
http://info.cern.ch/hypertext/WWW/Daemon/User/Config/Caching.html
・[Ref. 11]東田学, 対外接続のトラフィックを軽減するためのキャッシュ・サーバ導入について http://www.osaka-u.ac.jp/odins/document/proxy-jp.html
・[Ref. 12]佐藤豊、"多目的プロキシ・サーバ DeleGateの機能詳細"インターフェイス 1995年9月p130－p146
・[Ref. 13]インターナショナル・ビジネス・マシーンズ・コーポレイション、特開平6－309264、"キャッシュ・サーバ・ノードを有するコンピュータ・ネットワークにおける資源を探索する方法及び装置"
・[Ref. 14]"Siva——A hierarchical caching HTTP proxy"
http://www.cs.cornell.edu/Info/People/mdw/siva/siva.html
・[Ref. 15]"A Central Caching Proxy Server for WWW users at the University of Melbourne"
http://www.scu.edu.au/ausweb95/papers/indexing/ocallaghan/
・[Ref. 16]Lynx ブラウザについて
http://kufacts.cc.ukans.edu./about_lynx/about_lynx.html

【図面の簡単な説明】

【図１】本発明におけるゲートウェイ装置の一例であるｐｒｏｘｙサーバ計算機の動作概念図である。

【図２】WWWサーバ内の情報ページをクライアント計算機に読出したときの表示画面を示す図である。

【図３】実施の形態１のゲートウェイ装置であるｐｒｏｘｙサーバ計算機で行なわれるキャッシュ更新プロセスの処理手順を示すフローチャートである。

【図４】ｐｒｏｘｙプロセスにおけるキャッシュファイル管理の処理手順を示すフローチャートである。

【図５】実施の形態２のゲートウェイ装置であるｐｒｏｘｙサーバ装置で行なわれるキャッシュ更新プロセスの処理手順を示すフローチャートである。

【図６】実施の形態３のゲートウェイ装置であるｐｒｏｘｙサーバ装置で行なわれるキャッシュ更新プロセスの処理手順を示すフローチャートである。

【図７】本発明を実施した場合のサーバ計算機とクライアント計算機の間で転送されるデータのタイムチャートである。

【図８】実施の形態４における分散ファイルシステムの全体構成を示す図である。

【図９】実施の形態４におけるｒｅｄｉｒｅｃｔｏｒ計算機の処理手順を示すフローチャートである。

【図１０】図２に示す同一ページのファイルオブジェクトに対するアクセス要求が２台のサーバ計算機に分散されるようすを示す図である。

【図１１】実施の形態４におけるｒｅｄｉｒｅｃｔｏｒ計算機の内部構成を示す図である。

【図１２】実施の形態５におけるクライアント計算機の内部構成を示す図である。

【図１３】実施の形態６におけるｐｒｏｘｙサーバ計算機の内部構成を示す図である。

【図１４】実施の形態７における分散ファイルシステムの全体構成を示す図である。

【図１５】従来のゲートウェイ計算機の構成を示す図である。

【図１６】ｐｒｏｘｙサーバ装置の概念図である。

【図１７】ｐｒｏｘｙサーバ装置の回路構成の第１の例を示す図である。

【図１８】ｐｒｏｘｙサーバ装置の回路構成の第２の例を示す図である。

【図１９】ｐｒｏｘｙサーバ装置の回路構成の第３の例であって、本願発明の実施の形態でｐｒｏｘｙサーバ装置としても使用されるｐｒｏｘｙサーバ装置の回路構成を示す図である。

【図２０】キャッシュヒット率と平均アクセス速度との関係を示すグラフである。

【図２１】キャッシュファイルサイズとキャッシュヒット率との関係を示すグラフである。

【図２２】キャッシュ有効期限とキャッシュファイルの大きさとの関係を示すグラフである。

【図２３】従来の分散ファイルシステムの全体構成の第

57

１の例を示す図である。

【図２４】従来の分散ファイルシステムの全体構成の第２の例を示す図である。

【図２５】従来の分散ファイルシステムの全体構成の第３の例を示す図である。

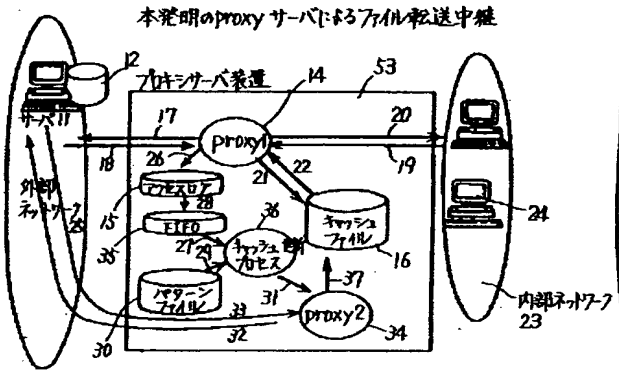
【図２６】従来の分散ファイルシステムにおけるサーバ計算機内のデータの更新とクライアント計算機が読出したデータとの関係を示すタイムチャートである。

【符号の説明】

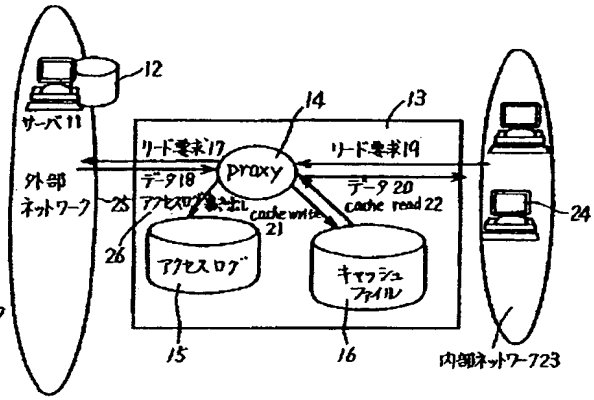
58

１１　サーバ計算機
１４　第１のｐｒｏｘｙプロセス
１５　アクセスログ
１６　キャッシュファイル
３０　パターンファイル
３４　第２のｐｒｏｘｙプロセス
３５　ＦＩＦＯバッファ
３６　キャッシュ更新プロセス
７２～７７　ｒｅｄｉｒｅｃｔｏｒ計算機

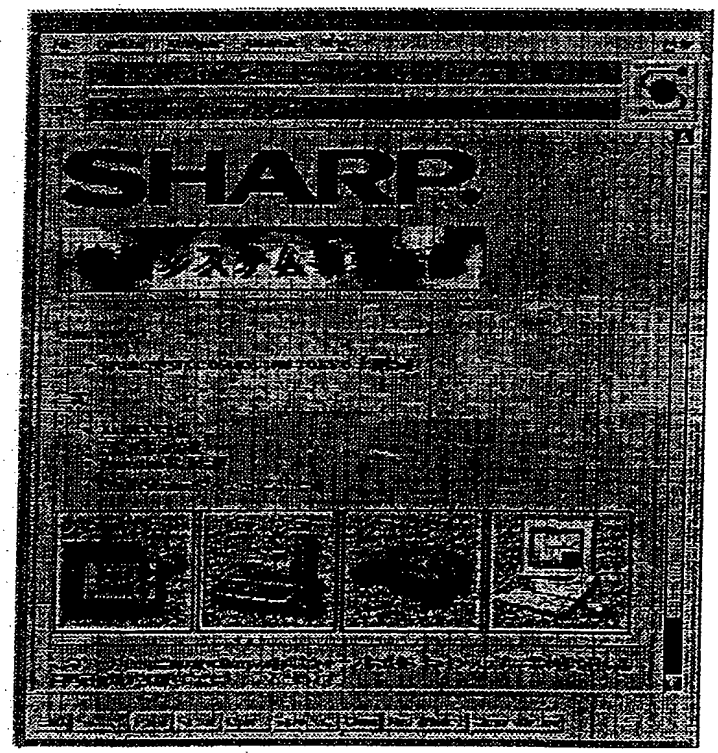
【図１】



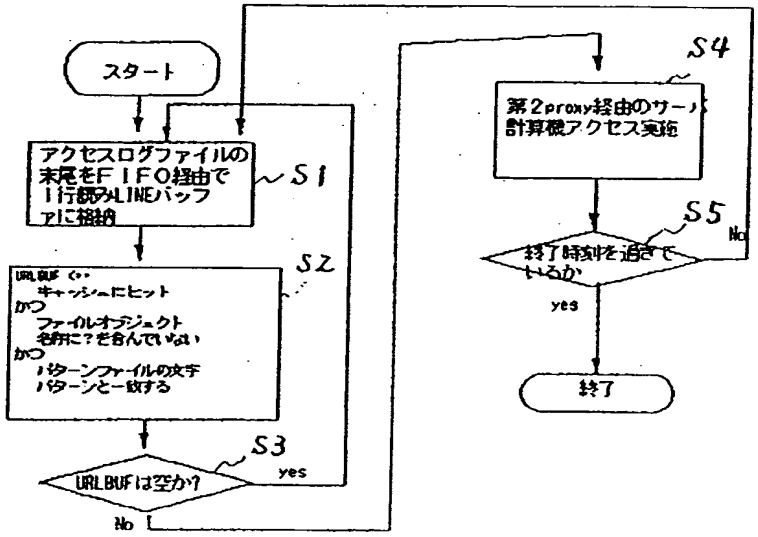
【図１６】



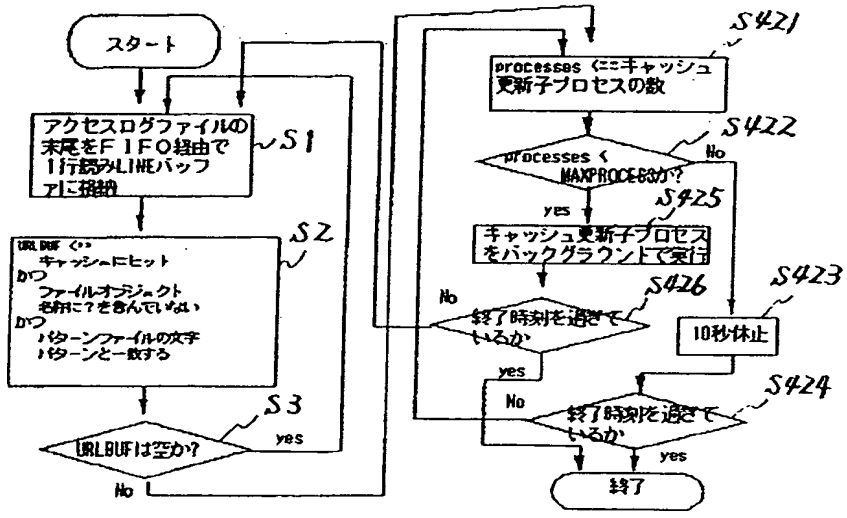
【図２】



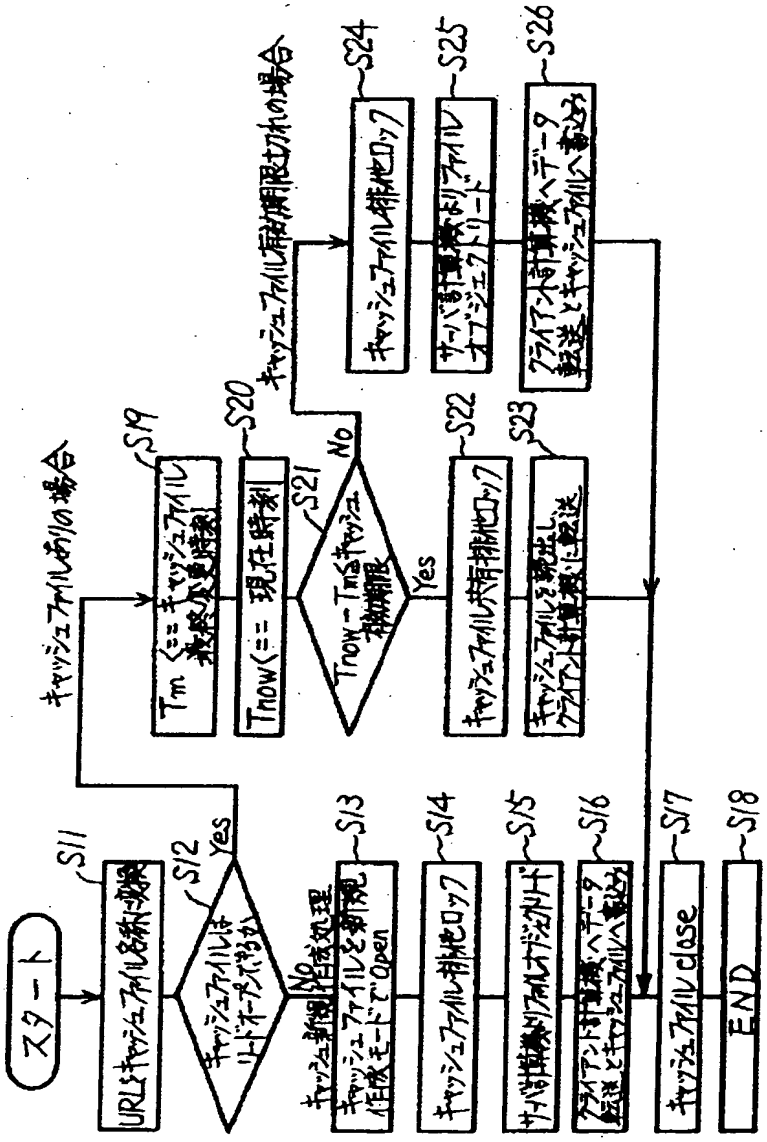
【図３】


スタート
アクセスログファイルの末尾をＦＩＦＯ経由で１行読みLINEバッファに格納
S1
URL BUF <>
キャッシュにヒット
かつ
ファイルオブジェクト名前に？を含んでいない
かつ
パターンファイルの文字パターンと一致する
S2
URLBUFは空か?
S3
yes
No
S4
第2proxy経由のサーバ計算機アクセス実施
S5
No
終了時刻を過ぎているか
yes
終了

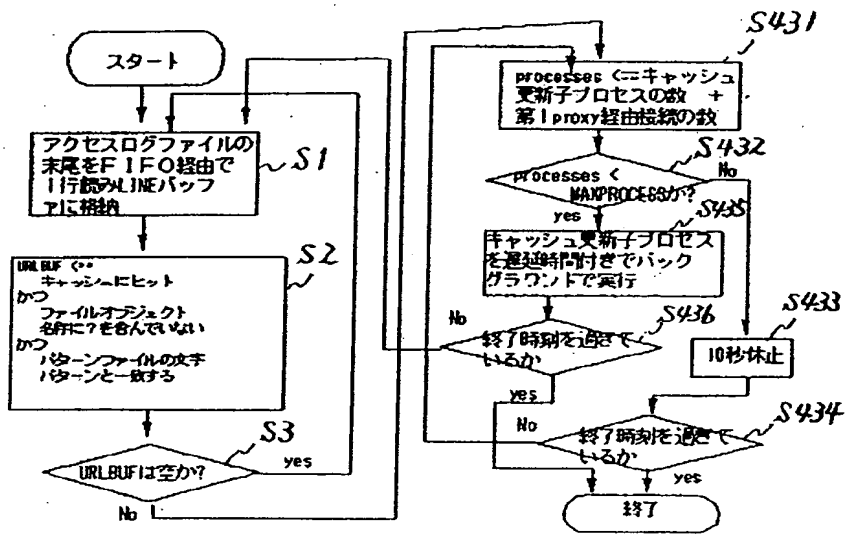
【図５】


スタート
アクセスログファイルの末尾をＦＩＦＯ経由で１行読みLINEバッファに格納
S1
URL BUF <>
キャッシュにヒット
かつ
ファイルオブジェクト名前に？を含んでいない
かつ
パターンファイルの文字パターンと一致する
S2
URLBUFは空か?
S3
yes
No
S421
processes <=キャッシュ更新子プロセスの数
S422
processes < MAXPROCESS か?
No
yes
S425
キャッシュ更新子プロセスをバックグラウンドで実行
S426
No
終了時刻を過ぎているか
yes
S423
10秒休止
S424
No
終了時刻を過ぎているか
yes
終了

【図４】

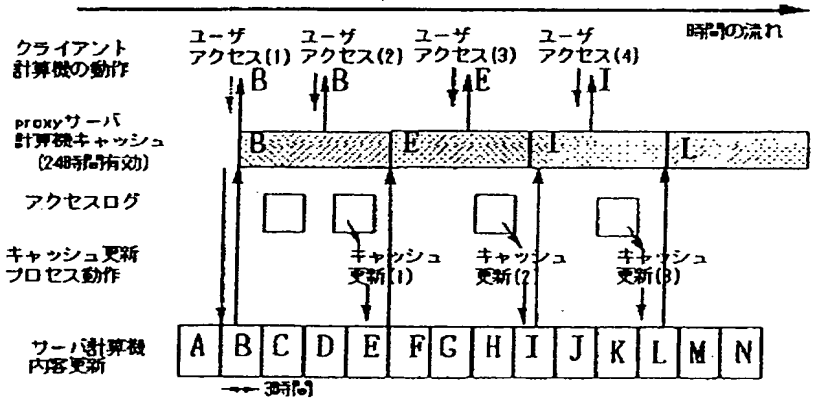

スタート
S11
URLとキャッシュファイル名称の取得
S12
キャッシュファイルは
リードオープンされた
Yes
No
S13
キャッシュファイルを新規
作成モードでOpen
S14
キャッシュファイル排他ロック
S15
サーバ時刻とファイルオブジェクト
S16
クライアント時刻へデータ
転送とキャッシュファイルへ書込み
S17
キャッシュファイルclose
S18
END
キャッシュファイルありの場合
S19
Tm ← キャッシュファイル
最終更新時刻
S20
Tnow ← 現在時刻
S21
Tnow - Tm ≦ キャッシュ
有効期限
No
Yes
S22
キャッシュファイル共有排他ロック
S23
キャッシュファイルを読出し
クライアント計算機へ転送
キャッシュファイル有効期限切れの場合
S24
キャッシュファイル排他ロック
S25
サーバ計算機よりファイル
オブジェクトをリード
S26
クライアント計算機へデータ
転送とキャッシュファイルへ書込み

【図６】

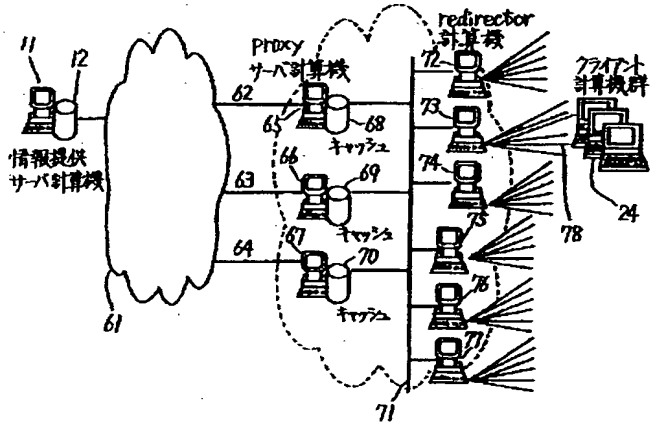


【図７】

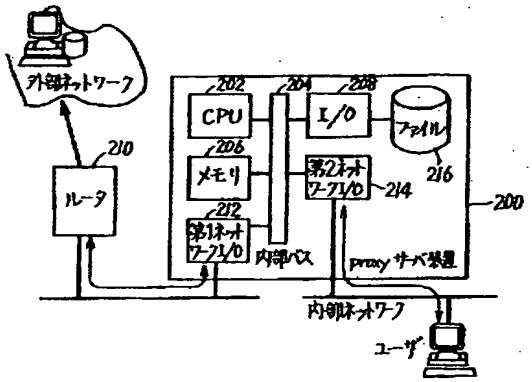
変更間隔3時間のサーバデータアクセスのタイムチャート（キャッシュ更新プロセスの動作時）



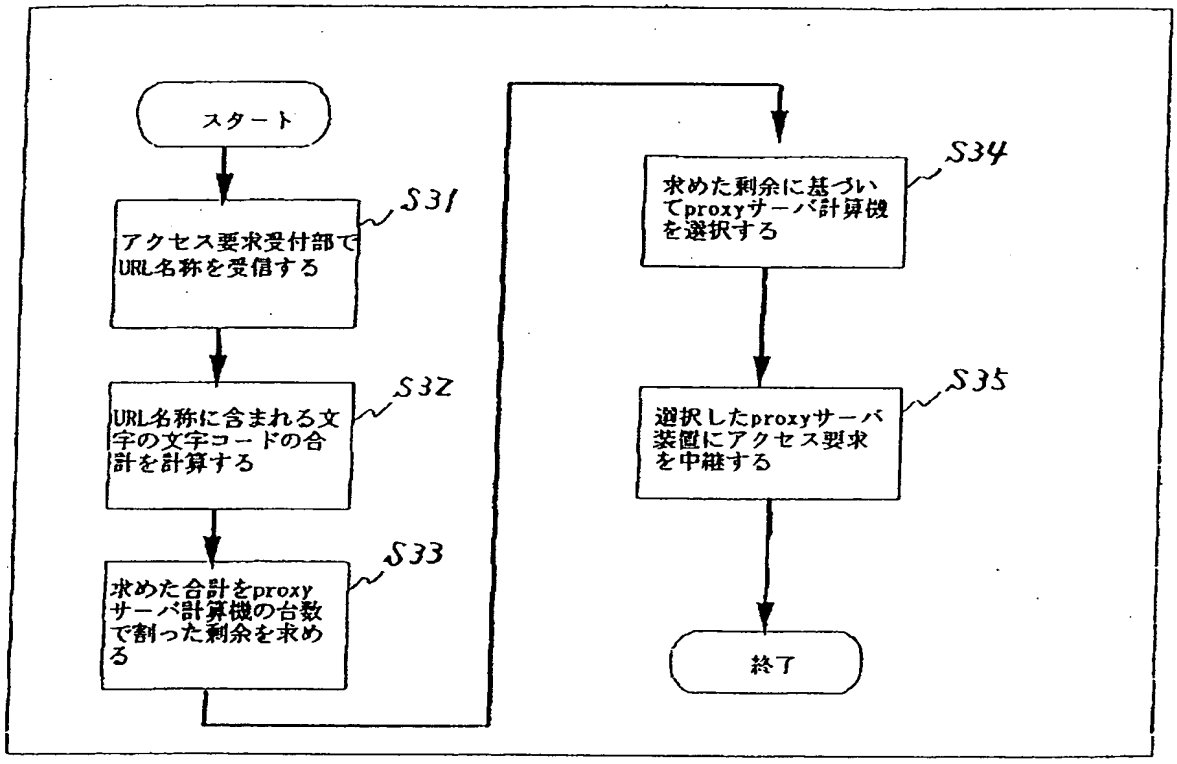
【図８】


11
12
情報提供サーバ計算機
61
Proxyサーバ計算機
62
63
64
65
66
67
68
キャッシュ
69
キャッシュ
70
キャッシュ
71
redirector計算機
72
73
74
75
76
77
78
クライアント計算機群
24

【図１７】


外部ネットワーク
210
ルータ
CPU
202
204
I/O
208
ファイル
216
メモリ
206
第2ネットワークI/F
214
第1ネットワークI/F
212
内部バス
200
proxyサーバ装置
内部ネットワーク
ユーザ

【図９】

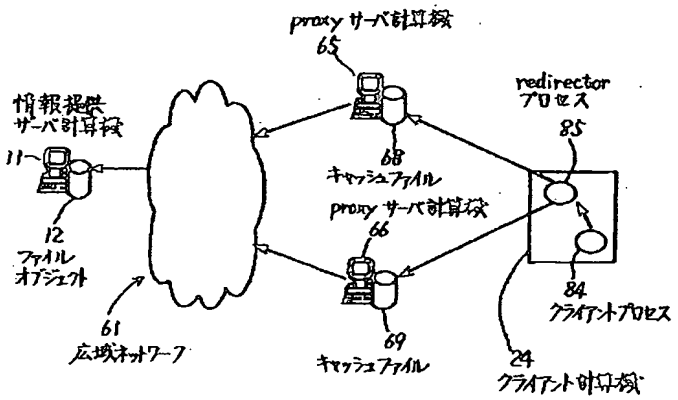

スタート
アクセス要求受付部でURL名称を受信する
S31
URL名称に含まれる文字の文字コードの合計を計算する
S32
求めた合計をproxyサーバ計算機の台数で割った剰余を求める
S33
求めた剰余に基づいてproxyサーバ計算機を選択する
S34
選択したproxyサーバ装置にアクセス要求を中継する
S35
終了

【図１０】

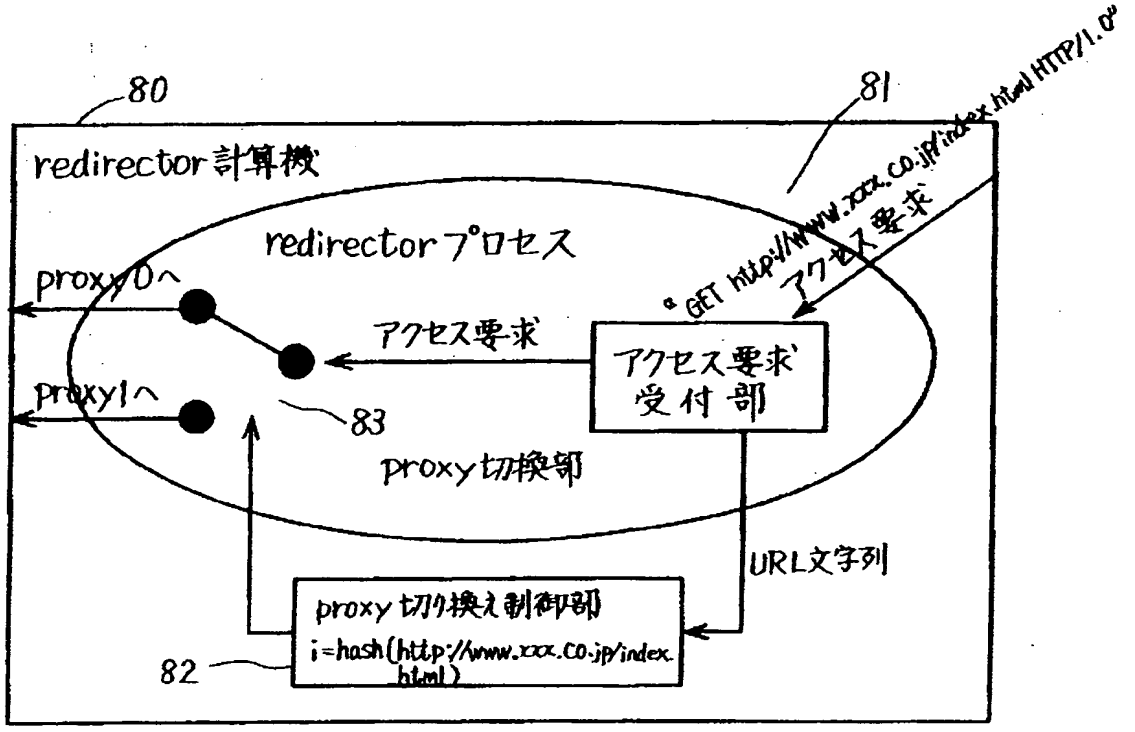
URL	check sum	checksumの2の剰余	選択されたproxyserver
http://naragw.sharp.co.jp/	2395	1	proxyserver1
http://naragw.sharp.co.jp/sharpcolor.gif	3836	0	proxyserver0
http://naragw.sharp.co.jp/isg.gif	3974	0	proxyserver0
http://naragw.sharp.co.jp/Zaurus.gif	3401	1	proxyserver1
http://naragw.sharp.co.jp/Shoin.gif	3264	0	proxyserver0
http://naragw.sharp.co.jp/Prostation.gif	3826	0	proxyserver0
http://naragw.sharp.co.jp/S2.gif	2884	0	proxyserver0

変換後のURL
http://proxyserver1:10080/-_-http://naragw.sharp.co.jp/
http://proxyserver0:10080/-_-http://naragw.sharp.co.jp/sharpcolor.gif
http://proxyserver0:10080/-_-http://naragw.sharp.co.jp/isg.gif
http://proxyserver1:10080/-_-http://naragw.sharp.co.jp/Zaurus.gif
http://proxyserver0:10080/-_-http://naragw.sharp.co.jp/Shoin.gif
http://proxyserver0:10080/-_-http://naragw.sharp.co.jp/Prostation.gif
http://proxyserver0:10080/-_-http://naragw.sharp.co.jp/S2.gif

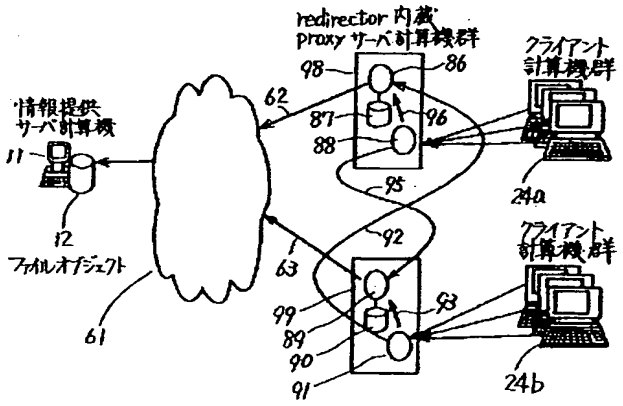
【図１２】



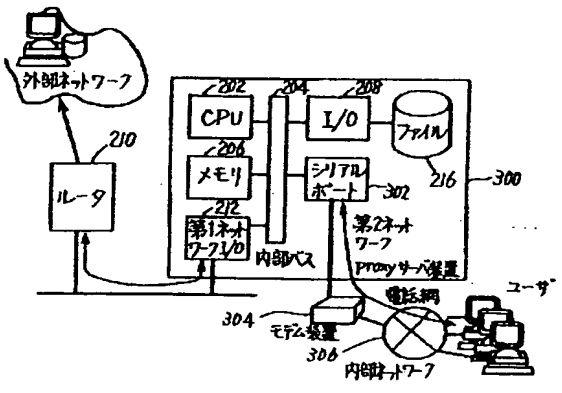
【図１１】


80
81
redirector計算機
redirectorプロセス
proxy0へ
proxy1へ
アクセス要求
アクセス要求受付部
83
proxy切換部
GET http://www.xxx.co.jp/index.html HTTP/1.0
アクセス要求
URL文字列
proxy切り換え制御部
i=hash(http://www.xxx.co.jp/index.html)
82

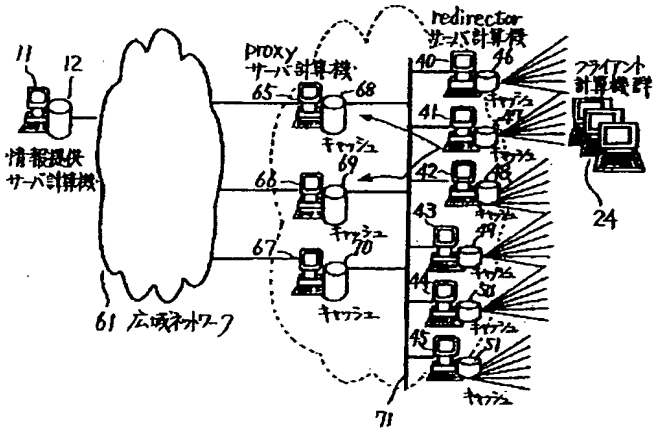
【図１３】


redirector内蔵
proxyサーバ計算機群
クライアント計算機群
情報提供サーバ計算機
11
12
ファイルオブジェクト
61
62
63
98
86
87
96
88
95
92
99
89
90
91
93
24a
24b

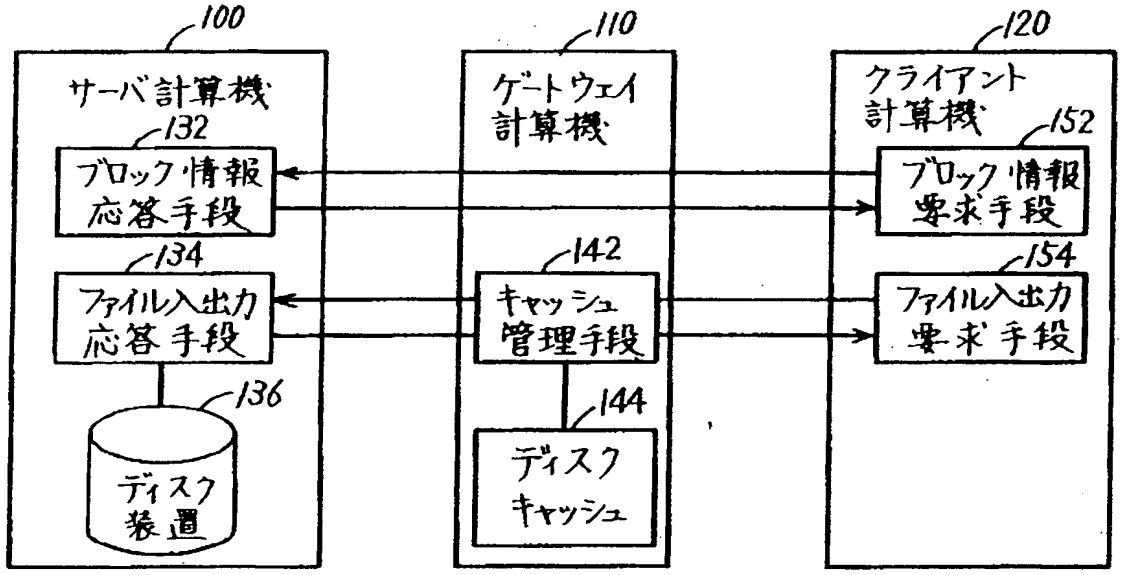
【図１８】


外部ネットワーク
210
ルータ
CPU
I/O
ファイル
メモリ
シリアルポート
第1ネットワークI/O
第2ネットワーク
内部バス
proxyサーバ装置
202
204
208
206
212
216
300
302
304
モデム装置
306
電話網
内部ネットワーク
ユーザ

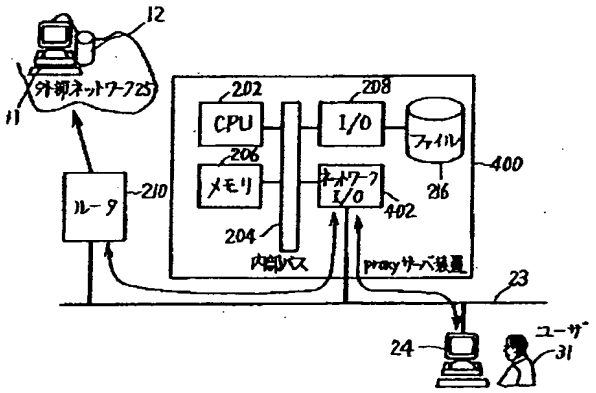
【図１４】


redirector
サーバ計算機
proxy
サーバ計算機
クライアント
計算機群
情報提供
サーバ計算機
キャッシュ
広域ネットワーク
11
12
24
40
41
42
43
44
45
61
65
66
67
68
69
70
71

【図１５】


100
サーバ計算機
132
ブロック情報
応答手段
134
ファイル入出力
応答手段
136
ディスク
装置
110
ゲートウェイ
計算機
142
キャッシュ
管理手段
144
ディスク
キャッシュ
120
クライアント
計算機
152
ブロック情報
要求手段
154
ファイル入出力
要求手段

【図１９】

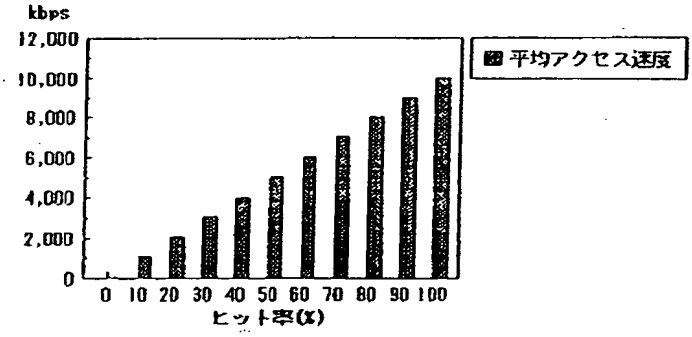


【図２０】

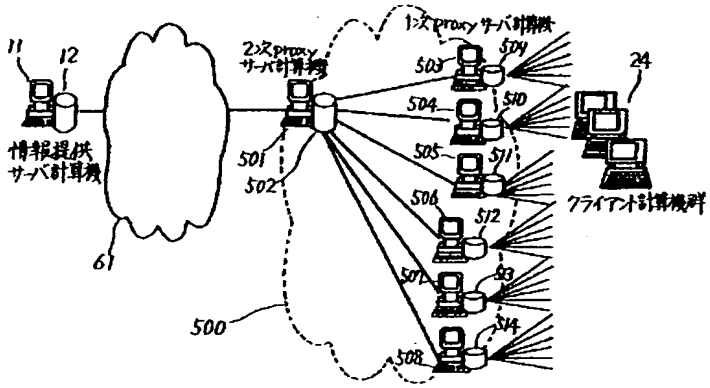
キャッシュヒット率向上による平均アクセス速度向上

- 平均アクセス速度 = 64kbps xミスヒット率 + 10Mbps x ヒット率

キャッシュヒット率と平均アクセス速度の関係

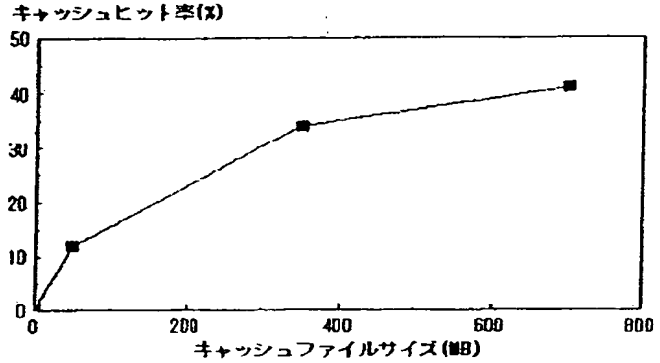


【図２３】



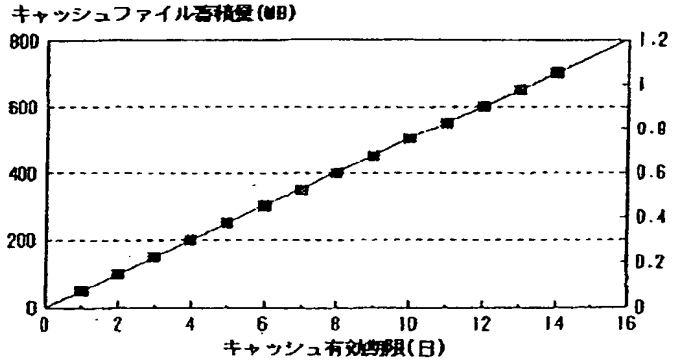
【図２１】

キャッシュファイルサイズと
キャッシュヒット率の関係

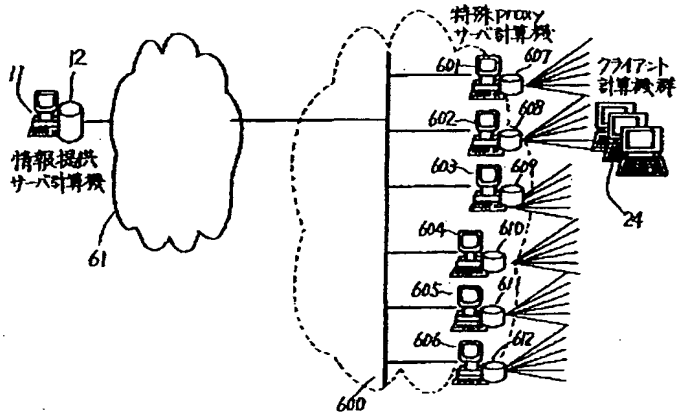


【図２２】

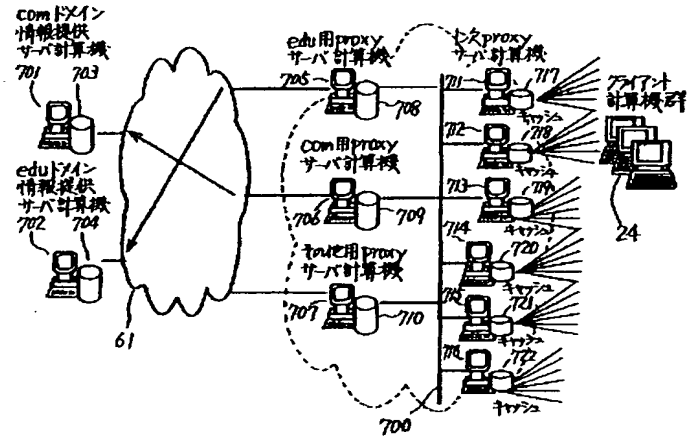
キャッシュ有効期限とキャッシュ
ファイルの大きさの関係



【図２４】


特殊proxy
サーバ計算機
クライアント
計算機群
情報提供
サーバ計算機
11
12
61
24
600
601
602
603
604
605
606
607
608
609
610
611
612

【図２５】


comドメイン
情報提供
サーバ計算機
eduドメイン
情報提供
サーバ計算機
edu用proxy
サーバ計算機
com用proxy
サーバ計算機
その他用proxy
サーバ計算機
その他proxy
サーバ計算機
クライアント
計算機群
キャッシュ
61
24
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722

【図２６】

変更間隔3時間のサーバデータアクセスのタイムチャート

- proxyサーバ計算機でのキャッシュは有効期限２４時間
- サーバ計算機のファイルオブジェクトは３時間間隔で更新